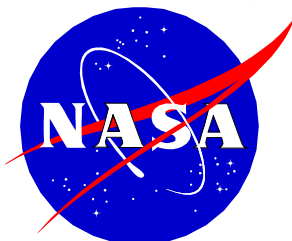


GMSEC Interface Specification

2016

March



Goddard Space Flight Center
Greenbelt, Maryland 20771

**National Aeronautics and
Space Administration**

GMSEC Interface Specification Document

2016
March

March 2016

Prepared by:

Matthew Handy
GMSEC API Lead Engineer
NASA/GSFC/ Code 583

Date

Approved by:

Dan Smith
GMSEC Project Manager
NASA/GSFC, Code 580

Date

Goddard Space Flight Center
Greenbelt, Maryland

Preface

This document is part of a series of versions of the Goddard Mission Services Evolution Center (GMSEC) Interface Specification Document. This document will continue to be updated and reissued as the GMSEC messages are defined.

GMSEC Mission Statement

The Goddard Mission Services Evolution Center (GMSEC) mission is to efficiently provide GSFC mission services. GMSEC is an integrated effort across multiple GSFC organizations providing mission enabling and cost and risk reducing data system solutions applicable to current and future NASA missions. GMSEC will enable the continued recognition of GSFC as a leader in space mission expertise and services.

Abstract

The GMSEC Interface Specification Document defines the standard messages used by GMSEC compliant software components in association with the GMSEC Applications Programming Interface (API) to achieve “plug and configure” compatibility and connectivity on the GMSEC Information Bus. The Information Bus is part of the GMSEC-defined architecture that describes an interconnection framework for building mission operations control centers and other mission-critical systems.

Document Evolution and Change History

This section provides a history of the changes to this document. With each released version of this document, additional GMSEC Message Definitions have been provided, or modifications to existing definitions have been updated.

The GMSEC Interface Specification Document has been developed in stages and released incrementally. The history of the document is given in the following table.

Table 0-1. Version History of the Interface Specification

Version Number	Release Description	Release Date
0.2	Demonstration 2 Baseline	2003 May 15
0.3	Demonstration 3 Baseline	2003 September 29
1.0	SMEX & TRMM Baseline	2004 April 15
1.1	Product Message Baseline	2004 June 30
1.2	Database Attributes Baseline	2004 September 3
1.3	Connectivity Ideas	2005 April
2005	Command Message Baseline	2005 November
2006 March	Field Standardization	2006 March
2007 May	Archive Mnemonic Stream of Messages	2007 May
2008 May	Updates to Field types and Messages	2008 June
2008 October	Message Exchange Patterns	2008 October
2009 May	Updates to Field Types and Messages	2009 May
2010 March	New Field Types in all messages to support 64-bit architectures	2010 March
2010 October	Modified for Generic Use	2010 October
2012 May	New messages: Telemetry TDM Frame Message, Processed Telemetry Message, Simple Service Message	2012 May
2013 February	Added User-Requested Changes to Messages	2013 February
2014 February	Added set of Navigation Data Messages	2014 February
2016 March	Update many data types in existing messages, normalize versioning information. See change log for complete details	2016 March

Table 0-2. Change Information History

Page #	Version	Change Date	Change Description
All	1.3	2005 April	All message fields with NUMBER-, NUMBER-OF-, NUM-, NUM-OF have been made consistent to NUM-OF-
38	2005	2005 Nov.	Table 4-4. Ordinal Date and Time Field Type Definition: Clarified that fields “YYYY” and “DDD” are fixed length, not variable.
40	2005	2005 Nov.	Information Bus Header: Changed “FACILITY” field to be Required (“R”)
Various	2005	2005 Nov.	For Archive Msg. Retrieval Req. & Resp., Archive Mnemonic Value Req. & Resp., Database Attrib. Req. & Resp., and all Product messages. Added product category and product information consistently. Also, modified Registration Req., Capability Req., and Capability Resp. messages accordingly.
148	2005	2005 Nov.	C2CX Heartbeat message: Added optional fields.
160	2005	2005 Nov.	Telem. Message CCSDS Packet: Changed LENGTH field type from “Int” to “Long”. Also, added values to QUALITY-CHECK fields.
Various	2005	2005 Nov.	For messages: Mnemonic Value Req., Archive Mnemonic Value Req., and Database Attributes Req. Changed Lower case “Mnemonic” to upper case “MNEMONIC”
Various	2005	2005 Nov.	For messages: Mnemonic Value Resp. and Mnemonic Value Data. Changed inconsistency of RAW-TLM-VALUE and RAW-VALUE to only use the latter. Same for EU-TLM-VALUE and EU-VALUE.
171, 202	2005	2005 Nov.	For messages: Replay Telem. Req. and Archive Mnemonic Value Req. Removed TIME-TYPE field.
251	2005	2005 Nov.	For Product Req. Message: Changed input parameters to “REQ-STRING”.

Page #	Version	Change Date	Change Description
All	2006	2006 March	Standardized field naming convention.
40	2007 May	2007 May	For GMSEC Information Bus Header , added fields "USERNAME" and "ROLE".
47	2007 May	2007 May	For Response Messages , added additional, optional status code of "6 – Final Message".
92	2007 May	2007 May	Added "System" category to component classifications.
Various	2007 May	2007 May	Changed incorrect use of the term RESPONSE-TYPE to RESPONSE-STATUS.
122	2007 May	2007 May	For Archive Message Retrieval Req. added fields DELIVER-VIA-REFERENCE and DELIVER-VIA INCLUDE to allow data product file to be included in response message.
126	2007 May	2007 May	For Archive Message Retrieval Resp. added DATA field to allow data product file to be included in response message.
144	2007 May	2007 May	For C2CX Control Message added CNTL-KEYWORD field for similarity with Directive Request Message.
146	2007 May	2007 May	Added new C2CX Device Status message
160, 161	2007 May	2007 May	For Telemetry Message added optional FINAL-MESSAGE field.
172	2007 May	2007 May	For Replay Telemetry Request Message , added NUM-OF-FILES and FILE.n.NAME-PATTERN so files could be specified for playback. Also, added PLAYBACK-RATIO as alternate method to specify playback speed. DATA-RATE is no longer required.
184	2007 May	2007 May	For Mnemonic Value Req. Message added MNEMONIC.n.DATA-TYPE, MNEMONIC.n.STATE-ATTRIBUTES, MNEMONIC.n.CRITERIA, and MNEMONIC.n.SAMPLE-RATE to be specified on a per mnemonic basis. Removed CRITERIA and SAMPLE-RATE from elsewhere in the message.

Page #	Version	Change Date	Change Description
190	2007 May	2007 May	For Mnemonic Value Response Message added TIME-COMPLETED field for consistency with other response messages.
Various	2007 May	2007 May	For messages: Archive Message Retrieval Req. and Resp. , Archive Mnemonic Value Req. and Resp. , Database Attributes Req. and Resp. , and Product Req, Msg, and Resp. : Changed PROD-SUBTYPE.n to PROD-SUBTYPE.n.NAME to be consistent with all other "NUM-OF-" naming conventions. See Section 4.1.1.1 Field Name.
206	2007 May	2007 May	For Archive Mnemonic Value Request Message added fields RESPOND-VIA-MSG, DELIVER-VIA-REFERENCE, DELIVER-VIA-INCLUDE, PLAYBACK-RATIO, and DATA-RATE to allow requested data to be 1) stream of messages, or 2) included in the Response message.
212	2007 May	2007 May	For Archive Mnemonic Value Response Message added DATA field to allow data to be included in the response message.
215	2007 May	2007 May	5.3.2.2.3 Archive Mnemonic Value Data Message . Added new message so behavior can mimic real-time Mnemonic Value Data Message.
Section 6	2007 May	2007 May	Removed Section 6 Concepts and Messages Under Consideration
Up front pages	2008 May	2008 May	Moved "Message Evolution" page and "Message Definition History" tables to Section 5.
36	2008 May	2008 May	Added new Field Type Definitions of Longlong, Quad, and ULonglong; added notes concerning Boolean and new field types
Appendix C	2008 May	2008 May	Added example of message use within a sample ground system architecture

Page #	Version	Change Date	Change Description
161	2008 May	2008 May	For Telemetry Message Contents of CCSDS Frame Message , defined Boolean values for RS-PRESENT field
54	2008 October	2008 Oct.	Added section on Message Exchange Patterns
144	2008 October	2008 Oct.	For C2CX Control Message added SPECIAL-INFO field for similarity with Directive Request Message.
146	2008 October	2008 Oct.	For C2CX Device Message, corrected DEVICE-STATUS and DEVICE-INFO fields to allow multiple devices.
251	2008 October	2008 Oct.	For Product Req. Message : added input file for additional optional parameters.
NA	2009 May	2009 May	Deleted Sections 3.6.1. and 3.6.2
36	2009 May	2009 May	Added new field types with explicit widths for unambiguous definitions across platforms.
122, 206, 228, and 251	2009 May	2009 May	For Archive Message Retrieval Request , Archive Mnemonic Value Request , Database Attributes Request , and Product Request messages changed field MAX-SIZE to SIZE for uniformity of structures.
251	2009 May	2009 May	For Product Request Message : added ability to include input files in message.
Various	2010 March	2010 March	All messages now specify the new unambiguous data types (e.g., F32, F64, I16, I32, I64, U16, and U32). Since the data type and field size have not changed, messages are not considered to have undergone a change. See Table 4-2. Field Type Definitions. IMPORANT NOTE: Applications that use

Page #	Version	Change Date	Change Description
			the new data types will not be able to compile or run with a GMSEC API prior to 3.0 since those APIs will not recognize the new data types.
Various	2010 March	2010 March	Files sizes were incorrectly specified in some messages as short rather than long. All fields that specify file sizes are l32. Messages affected are Archive Message Retrieval Response , Product Request , Product Response , and Product Message .
40	2010 March	2010 March	Added optional field, MESSAGE-CLASS, to GMSEC Information Bus Header.
122	2010 March	2010 March	For Archive Message Retrieval Request , field MSG-ID is now required.
151	2010 March	2010 March	For C2CX RSRC (Resource) Message , added logical IP and physical EUI address identifiers to network port description.
172	2010 March	2010 March	For Replay Telemetry Request Message , added COLLECTION-POINT and ORBIT fields to further distinguish data requests. Also, STREAM-MODE field value of "RT" can be used to request real-time data streams.
184	2010 March	2010 March	For Mnemonic Value Request Message , added START-TIME and STOP-TIME fields as another data selection criterion.
195, 219, and 257	2010 March	2010 March	For Mnemonic Value Data Message , Archive Mnemonic Value Data Message , and Product Message added optional field, FINAL-MESSAGE, to note last message in series.
Various	2011 February	2011 February	Removed mission and satellite specific information.
163 and 164	2012 May	2012 May	Added Telemetry TDM Frame Message and Processed Telemetry Message to the set of Telemetry Messages

Page #	Version	Change Date	Change Description
160-164, 172, 184, 206, 328	2012 May	2012 May	For all Real-Time Telemetry Data Messages , and the Replay Telemetry Request Message , Mnemonic Value Request Message , and Archive Mnemonic Value Request Message added optional field COLLECTION-POINT for consistency with Replay Telemetry Request Message (see above). Also, added COLLECTION-POINT as optional <i>ME6</i> subject element.
244 and 247	2012 May	2012 May	For Command Request and Command Response Messages , added optional RELEASE-TIME field.
265	2012 May	2012 May	Added pair of Simple Service Request and Simple Service Response messages.
293	2012 May	2012 May	Revised materials on the use of services in the “Sample” Appendix C which became Section 6 “GMSEC Services”. Section 6 on the GMSEC API became Appendix C.
Various	2013 January	2013 January	For all messages, changed MSG-ID field data type from String to Header String to ensure proper use in message subject element. CONTENT-VERSION is unchanged.
Section 4.2 and 4.3	2013 January	2013 January	Modified description on message exchange pattern behavior for new API function “Request/Multiple-Reply”
137	2013 January	2013 January	For message subject of Directive Request Message added optional ME2: DIRECTIVE-KEYWORD from message content. CONTENT-VERSION is unchanged.
153	2013 January	2013 January	For message subjects of C2CX Messages added optional ME6: CNTL-KEYWORD from message content. CONTENT-VERSION is unchanged.
146	2013 January	2013 January	For C2CX Device Message added DEVICE.n.PARAM.n.TIME. Changed DEVICE.n.PARAM.n.VALUE from Float to variable data type. New CONTENT-VERSION.
148	2013 January	2013 January	For C2CX Heartbeat Message changed SW-VERSION from Float to variable data type. Component must determine data type before accessing. New CONTENT-VERSION.

Page #	Version	Change Date	Change Description
161, 163, and 164	2013 January	2013 January	For Telemetry Messages for CCSDS Frame, TDM, and Processed Telemetry Frame , added optional FRAMESYNC-STATUS field. New CONTENT-VERSION.
164	2013 January	2013 January	For Processed Telemetry Frame , changed name of FORMAT.n.IDENTIFIER field. New CONTENT-VERSION.
251	2013 January	2013 January	For Product Request Message , added missing field names in table. CONTENT-VERSION is unchanged.
Various	2013 February	2013 February	For Archive Message Retrieval Request Message, Archive Message Retrieval Response Message, Archive Mnemonic Value Request Message, Archive Mnemonic Value Response Message, Database Attributes Request Message, Database Attributes Response Message for List All Mnemonics, Product Request Message, Product Response Message, and Product Message , added PROD-DESCRIPTION and FILE.n.DESCRPTION fields. New CONTENT-VERSION.
54 and following	2014 February	2014 February	Corrections made to Message Exchange Pattern section.
277	2014 February	2014 February	Added set of Navigation Data Messages
Various	2016 March	2016 March	Changed all values which can only be positive numbers to be unsigned. Removed requirement for specifying MSG-ID. Removed requirement for specifying LENGTH of binary arrays as separate field. Added Main Memory information to resource messages. Message version table contains complete change set.

Table of Contents

SECTION 1 INTRODUCTION.....	1
1.1 BACKGROUND.....	1
1.2 OVERVIEW	1
1.3 SCOPE OF THE INTERFACE SPECIFICATION	2
1.3.1 File Transfers	2
1.3.2 GMSEC Compliant Components.....	3
1.4 APPLICABLE AND REFERENCED DOCUMENTS	3
1.4.1 GMSEC Documents	3
1.4.2 Existing Space Data Systems Standards	3
1.4.2.1 Telemetry and Command Data References	3
1.4.2.2 XML Telemetric and Command Exchange (XTCE) Reference..	3
1.4.3 Potential Future Space Data Systems Standards	4
1.4.3.1 Ground Operations Automation Language.....	4
1.4.3.2 Spacecraft Monitor and Control.....	5
1.4.4 Document Creation Tools.....	5
SECTION 2 QUALITY OF COMMUNICATIONS SERVICE.....	7
SECTION 3 GMSEC MESSAGE SUBJECT DEFINITION	11
3.1 CHARACTERISTICS OF GMSEC MESSAGE SUBJECT NAMES	12
3.2 FORMAT OF GMSEC MESSAGE SUBJECTS (OR TOPICS)	13
-3.3 STANDARDS OF GMSEC MESSAGE SUBJECT NAMES.....	16
3.3.1 Subject Standard Element of the GMSEC Message Subject.....	18
3.3.2 Mission Elements of the GMSEC Message Subject	18
3.3.3 Message Elements of the GMSEC Message Subject.....	20
3.3.4 Miscellaneous Elements of the GMSEC Message Subject.....	21
3.3.3.1 Service Disposition	22
3.4 DEFINITIONS OF GMSEC SUBJECT NAMES	24
3.5 EXAMPLE OF THE GMSEC MESSAGE SUBJECT	27
SECTION 4 GMSEC STANDARD MESSAGE DEFINITION	31
4.1 FORMAT OF MESSAGE DEFINITIONS	31
4.1.1 General Message Format	31
4.1.1.1 Field Name	32
4.1.1.2 Required, Optional, and API Field Indicator	34
4.1.1.3 Value.....	34
4.1.1.4 Type.....	34
4.1.2 GMSEC Information Bus Header	40
4.1.2.1 Notes on Fields in the Information Bus Header	41
4.1.2.1.1 Fields Used for Message Subjects	41

4.1.2.1.2 Fields for Message Uniqueness.....	42
4.1.2.1.3 Middleware Tracking Information	42
4.1.2.1.4 Component Information – Location and ID	42
4.1.2.1.5 Component Information – Logical	43
4.1.3 Message Contents.....	43
4.1.3.2 Other Fields in the Message Content Portion.....	44
4.2 GMSEC MESSAGES: THEIR CHARACTERISTICS AND INTERACTIONS.....	45
4.2.1 GMSEC Message Type Overview.....	45
4.2.1.1 GMSEC MSG Message Type	46
4.2.1.2 GMSEC REQ Message Type	46
4.2.1.3 GMSEC RESP Message Type	47
4.2.1.5 GMSEC API Transport Mechanisms	50
4.2.1.5.1 GMSEC API Send and Receive Functions	50
4.2.1.5.2 Behavior of the GMSEC API Send and Receive Functions	51
4.3 GMSEC MESSAGE EXCHANGE PATTERNS	54
4.3.1 MEP Publish.....	58
4.3.1.1 Description	58
4.3.1.2 Usage	58
4.3.1.3 Sequence Diagram	58
4.3.2 MEP Request/Response (“Variations on a Theme”)	59
4.3.2.1 MEP Request/ACK.....	59
4.3.2.1.1 Description	59
4.3.2.1.2 Usage	59
4.3.2.1.3 Sequence Diagram	59
4.3.2.2 MEP Request/Response	60
4.3.2.2.1 Description	60
4.3.2.2.2 Usage	60
4.3.2.2.3 Sequence Diagram	60
4.3.2.3 MEP Request/ACK/Response.....	61
4.3.2.3.1 Description	61
4.3.2.3.2 Usage	61
4.3.2.3.3 Sequence Diagram	61
4.3.2.4 MEP Request/ACK/Interim Status/Response.....	62
4.3.2.4.1 Description	62
4.3.2.4.2 Usage	62
4.3.2.4.3 Sequence Diagram	63
4.3.2.5 MEP Request/Interim Status/Response	64
4.3.2.5.1 Description	64
4.3.2.5.2 Usage	64
4.3.2.5.3 Sequence Diagram	64
4.3.3 MEP Message Triad	65
4.3.3.1 MEP Triad 1: Request/Response/Publish	66
4.3.3.1.1 Description	66
4.3.3.1.2 Usage	66
4.3.3.1.3 Sequence Diagram	67

4.3.3.2 MEP Triad 2: Publish/Request/Response	68
4.3.3.2.1 <i>Description</i>	68
4.3.3.2.2 <i>Usage</i>	68
4.3.3.2.3 <i>Sequence Diagram</i>	69
4.3.4 MEP Subscription	70
4.3.4.1 <i>Description</i>	70
4.3.4.2 <i>Usage</i>	70
4.3.4.3 <i>Sequence Diagram</i>	70
4.3.5 CCSDS Spacecraft Monitor and Control - Message Abstraction Layer.....	72
4.4 PRODUCT MESSAGES	75
4.4.1 GMSEC and Component Handling.....	75
4.4.1.1 <i>Background and Scope</i>	75
4.4.1.2 <i>Product Generation and Distribution Factors.....</i>	75
4.4.1.3 <i>GMSEC Product Message Summary</i>	76
4.4.2 Product Characteristics	77
4.4.2.1 <i>Production</i>	77
4.4.2.1.1 <i>Methods of Production</i>	77
4.4.2.1.2 <i>GMSEC Facilitation.....</i>	77
4.4.2.2 <i>Content and Format</i>	83
4.4.2.2.1 <i>Forms of Products</i>	83
4.4.2.2.2 <i>Product Attributes or Properties.....</i>	83
4.4.2.2.3 <i>GMSEC Facilitation.....</i>	83
4.4.2.3 <i>Distribution</i>	85
4.4.2.3.1 <i>Product Generation and Distribution.....</i>	85
4.4.2.3.2 <i>GMSEC Facilitation.....</i>	85
4.4.3 Product Message Examples	88
4.5 CATEGORIZATION	89
4.5.1 Message Categories and Message Overview.....	89
4.5.1.1 <i>Control and Monitor</i>	91
4.5.1.2 <i>Data Messages</i>	91
4.5.1.3 <i>Product and Service Messages</i>	92
4.5.2 Component Categories.....	93
4.5.3 Product Categories.....	94
SECTION 5 GMSEC STANDARD MESSAGES	99
5.1 MESSAGE HISTORY	99
5.1.1 Message Evolution Pages	99
5.1.2 Message Definition History	100
5.2 CONTROL AND MONITOR LEVEL MESSAGES.....	105
5.2.1 General Information and Notification Messages	105
5.2.1.1 Log Message	105
5.2.1.1.1 <i>Log Message Information Bus Header.....</i>	106
5.2.1.1.2 <i>Log Message Contents.....</i>	107
5.2.1.1.3 <i>Log Message Subjects.....</i>	114
5.2.1.2 Archive Message Retrieval.....	116

5.2.1.2.1 Archive Message Retrieval Request	116
5.2.1.2.2 Archive Message Retrieval Response	124
5.2.1.2.3 Archive Message Retrieval Subjects	128
5.2.2 Action and Solicitation Messages	130
5.2.2.1 Directive Messages	132
5.2.2.1.1 Directive Request Message	132
5.2.2.1.2 Directive Response Message	135
5.2.2.1.3 Directive Message Subjects	137
5.2.2.2 Component-To-Component Transfer (C2CX) Messages	139
5.2.2.2.1 Component-To-Component Transfer Message Information Bus Header	140
5.2.2.2.2 Component-To-Component Transfer Message Content	140
5.2.2.2.3 Component-to-Component Transfer Message Subjects	153
5.3 DATA LEVEL MESSAGES	156
5.3.1 Telemetry Data Messages	156
5.3.1.1 Real-Time Telemetry Data Messages	158
5.3.1.1.1 Real-Time Telemetry Data Messages Information Bus Header	159
5.3.1.1.2 Real-Time Telemetry Data Message Contents	160
5.3.1.1.3 Real-Time Telemetry Data Message Subjects	167
5.3.1.2 Replay Telemetry Data Messages	169
5.3.1.2.1 Replay Telemetry Data Request Message	170
5.3.1.2.2 Replay Telemetry Data Response Message	175
5.3.1.2.3 Replay Telemetry Data Message Subjects	177
5.3.2 Mnemonic Data Messages	180
5.3.2.1 Real-Time Mnemonic Value Messages	180
5.3.2.1.1 Mnemonic Value Request Message	180
5.3.2.1.2 Mnemonic Value Response Message	187
5.3.2.1.3 Mnemonic Value Data Message	193
5.3.2.1.4 Mnemonic Value Data Message, Request, and Response Message Subjects	199
5.3.2.2 Archive Mnemonic Value Messages	202
5.3.2.2.1 Archive Mnemonic Value Request Message	202
5.3.2.2.2 Archive Mnemonic Value Response Message	210
5.3.2.2.3 Archive Mnemonic Value Data Message	215
5.3.2.2.4 Archive Mnemonic Value Message Subjects	222
5.3.2.3 Database Attributes Messages	225
5.3.2.3.1 Database Attributes Request Message	225
5.3.2.3.2 Database Attributes Response Message	230
5.3.2.3.3 Database Attributes Message Subjects	239
5.3.3 Satellite Command Messages	241
5.3.3.1 Command Request Message	241
5.3.3.1.1 Command Request Message Information Bus Header	243
5.3.3.1.2 Command Request Message Contents	244
5.3.3.2 Command Response Message	245

5.3.3.2.1 Command Response Message Information Bus Header.....	246
5.3.3.2.2 Command Response Message Contents.....	247
5.3.3.3 Command Message Subjects.....	248
5.4 PRODUCTS AND SERVICES LEVEL MESSAGES.....	250
5.4.1 Product Messages	250
5.4.1.1 Product Request Message	250
5.4.1.1.1 Product Request Message Information Bus Header.....	250
5.4.1.1.2 Product Request Message Content.....	251
5.4.1.2 Product Response Message	253
5.4.1.2.1 Product Response Message Information Bus Header.....	253
5.4.1.2.2 Product Response Message Contents.....	254
5.4.1.3 Product Message.....	256
5.4.1.3.1 Product Message Information Bus Header.....	256
5.4.1.3.2 Product Message Contents.....	258
5.4.1.4 Product Request, Product Response, and Product Message Subjects	260
5.4.2 Simple Service Messages.....	265
5.4.2.1 Simple Service Request Message.....	266
5.4.2.1.1 Simple Service Request Message Information Bus Header	267
5.4.2.1.2 Simple Service Request Message Contents.....	268
5.4.2.2 Simple Service Response Message.....	270
5.4.2.2.1 Simple Service Response Message Information Bus Header	271
5.4.2.2.2 Simple Service Response Message Contents.....	271
5.4.2.3 Simple Service Message Subjects.....	272
5.5 NAVIGATION DATA MESSAGES.....	277
5.5.1 Attitude Data Messages	279
5.5.1.1 Attitude Parameter Message (APM) Contents	279
5.5.1.2 Attitude Ephemeris Message (AEM) Contents.....	281
5.5.2 Orbit Data Messages	283
5.5.2.1 Orbit Parameter Message (OPM) Contents	283
5.5.2.2 Orbit Mean-Elements Message (OMM) Contents.....	285
5.5.2.3 Orbit Ephemeris Message (OEM) Contents	287
5.5.3 Tracking Data Messages (TDM)	289
5.5.3.1 Tracking Data Message (TDM) Contents.....	289
6 GMSEC SERVICES.....	293
6.1 SERVICES WITHIN THE GMSEC ARCHITECTURE	293
6.1.1 Description	293
6.1.2 Facets of the GMSEC Software Information Bus.....	295
6.1.3 Some Differences between GMSEC Services and other Architectures.....	296
6.1.4 Creating a Service within the GMSEC Architecture.....	301
6.1.4.1 Using the <i>ME1</i> Element of the Message Subject.....	302
6.2 USE OF GMSEC SERVICES WITHIN A SAMPLE GROUND SYSTEM ARCHITECTURE	303
6.1.1 Flight Dynamics Subsystem	305
6.1.2 Planning and Scheduling Subsystem	307

6.1.3 Archive and Assessment Subsystem	309
6.1.4 Telemetry and Command Subsystem	311
6.1.5 Front End Processors and Simulators Subsystem	313
6.1.6 Automation Subsystem	314
6.1.7 Common Services across Subsystems	315
APPENDIX A ACRONYMS.....	317
APPENDIX B GMSEC COMMON MESSAGE STRUCTURES.....	319
B.1 STRUCTURE: BODY CONTENT IDENTIFICATION	321
B.2 STRUCTURE: BODY CONTENT SUBTYPE	321
B.3 STRUCTURE: DATA ABSTRACT	322
B.4 STRUCTURE: DATA FILE	322
B.5 STRUCTURE: DIRECTIVE KEYWORD.....	323
B.6 STRUCTURE: DIRECTIVE AND SERVICE PROCESSING DIRECTIONS.....	323
B.7 STRUCTURE: FILE ATTRIBUTES.....	324
B.8 STRUCTURE: MNEMONIC SELECTION CRITERIA.....	324
B.9 STRUCTURE: PRODUCT DISTRIBUTION OPTIONS	325
B.10 STRUCTURE: PRODUCT CATEGORY IDENTIFICATION	325
B.11 STRUCTURE: RESPONSE STATUS	326
B.12 STRUCTURE: SATELLITE COMMAND DESCRIPTION	327
B.13 STRUCTURE: SATELLITE COMMAND EXECUTION TIME PARAMETERS	327
B.14 STRUCTURE: SATELLITE COMMAND PARAMETERS.....	327
B.15 STRUCTURE: TELEMETRY DATA STREAM INFORMATION	328
B.16 STRUCTURE: TELEMETRY DATA CHANNEL INFORMATION.....	328
B.17 STRUCTURE: TIME WINDOW	329
APPENDIX C GMSEC APPLICATION PROGRAMMING INTERFACE.....	331
C.1 MESSAGING OVERVIEW.....	331
C.1.1 Publish / Subscribe	332
C.1.2 Request / Reply	333
C.1.3 Persistence	334
C.2 APPLICATION PROGRAMMING INTERFACE (API).....	334
C.2.1 API Function Descriptions	335
C.3 ANCILLARY PROGRAMMING INFORMATION.....	337
C.3.1 Programming Languages and Platforms Supported.....	337
C.3.2 Libraries.....	338
C.3.3 Building and Linking Applications	338
APPENDIX D FUTURE CONSIDERATIONS	339
APPENDIX E CHECKLIST FOR UPDATING THIS DOCUMENT.....	341

List of Figures and Tables

Table 0-1. Version History of the Interface Specification	vi
Table 0-2. Change Information History.....	vii
Table 2-1. Communication Model Comparison	7
Table 2-2. Speed and Quality of Delivery Mechanisms.....	10
Table 3-1. Sample Subject Filtering Items in Addition to Type and Subtype	12
Table 3-2. Subject Matching Examples	15
Table 3-3. GMSEC Message Subject Name Definition.....	17
Table 3-4. Subject Standard Element of the GMSEC Message Subject.....	18
Table 3-5. Mission Elements of the GMSEC Message Subject.....	18
Table 3-6. Descriptions of the Mission Elements of the GMSEC Message Subject	18
Table 3-7. Message Elements of the GMSEC Message Subject	20
Table 3-8. Descriptions of the Message Elements of the GMSEC Message Subject.....	20
Table 3-9. <i>Miscellaneous Elements</i> of the GMSEC Message Subject.....	21
Table 3-10. Message Type Determines Content of the <i>Miscellaneous Elements</i>	21
Table 3-11. Message Subject Syntax	24
Table 3-12. GMSEC Message Subject Definitions, 1 of 4	24
Table 3-13. GMSEC Message Subject Definitions, 2 of 4	25
Table 3-14. GMSEC Message Subject Definitions, 3 of 4	26
Table 3-15. GMSEC Message Subject Definitions, 4 of 4	26
Table 3-16. Directive Request Message Subject Naming	27
Table 3-17. Properties of the <i>Miscellaneous Elements</i> for the Directive Request Message	27
Table 3-18. Directive Response Message Subject Naming	28
Table 3-19. Properties of the <i>Miscellaneous Elements</i> for the Directive Response Message	28
Table 4-1. General Message Description Format	32
Table 4-2. Field Type Definitions	36
Table 4-3. Ordinal Date and Time Field Type Definition	38
Table 4-4. GMSEC Information Bus Header	40
Table 4-5. Fixed Portion of the GMSEC Message Subject.....	41
Table 4-6. Mapping of Information Bus Header Fields to the GMSEC Message Subject.....	42
Table 4-7. Format of the Message Content Portion.....	43
Table 4-8. GMSEC Message Subject Name Definition	45
Table 4-9. Response Status Substructure	47
Table 4-10. Sequence of Response Status	49
Table 4-11. API Send Functions Used with the GMSEC Message Types	50
Table 4-12. API Receive Functions Used with the GMSEC Message Types	50
Figure 4.2.1.5.2.1-1 Sequence Diagram of Request-Response Message Exchange Using the API Request and Reply Functions.....	52
Table 4-13. GMSEC Message Exchange Pattern Types.....	54
Table 4-14. GMSEC Message Exchange Patterns	55
Figure 4.3.1.3-1 Sequence Diagram of Publish Message Exchange Pattern	58
Figure 4.3.2.1.3-1 Sequence Diagram of Request/ACK Message Exchange Pattern	59
Figure 4.3.2.2.3-1 Sequence Diagram of Request/Response Message Exchange Pattern	60
Figure 4.3.2.3.3-1 Sequence Diagram of Request/ACK/Response Message Exchange Pattern	61
Figure 4.3.2.4.3-1 Sequence Diagram of Request/ACK/Interim Status/Response Message Exchange Pattern.....	63
Figure 4.3.2.5.3-1 Sequence Diagram of Request/Interim Status/Response Message Exchange Pattern	64
Figure 4.3.3.1.3-1 Sequence Diagram of Triad 1: Request/Response/Publish Message Exchange Pattern	67

Figure 4.3.3.2.3-1 Sequence Diagram of Triad 2: Publish/Request/Response Message Exchange Pattern	69
Figure 4.3.4.3-1 Sequence Diagram of Subscription Message Exchange Pattern	71
Table 4-15. MAL Interaction Patterns with Associated Interaction Pattern Stages	73
Table 4-16. Comparison of SM&C and GMSEC Interaction Patterns	74
Table 4-17. Uses of the Product Message	76
Table 4-18. Relationship of Product Generation and Distribution	85
86	
Figure 4.4.2.3.2-1 Product Distribution Options	86
Table 4-19. Product Generation and Distribution Scenarios	87
Figure 4.4.3-1 Requests for Data Plot Using GMSEC Messages	88
Figure 4.4.3-2 Responses to Data Plot Request Using GMSEC Messages	88
Table 4-20. GMSEC Message Functional Grouping	90
Table 4-21. Software Class and Subclass Categories	93
Table 4-22. Product Categories	94
Table 5-1. GMSEC Message Definitions History, 1 of 4	100
Table 5-2. GMSEC Message Definitions History, 2 of 4	102
Table 5-3. GMSEC Message Definitions History, 3 of 4	103
Table 5-4. GMSEC Message Definitions History, 4 of 4	104
Table 5-5. Log Message Summary	105
Table 5-6. Log Message Information Bus Header	106
Table 5-7. Log Message Contents	107
Table 5-8. Pass-Related Occurrence Types	109
Table 5-9. Telemetry Limit Violation Occurrence Types	110
Table 5-10. Command Verification Occurrence Types	111
Table 5-11. Miscellaneous Occurrence Types	112
Table 5-12. Log Message Subject Naming	114
Table 5-13. Properties of the <i>Miscellaneous Elements</i> for the Log Message	114
Table 5-14. Archive Message Retrieval Request Summary	117
Table 5-15. Examples of Time Fields in GMSEC Messages, 1 of 2	119
Table 5-16. Examples of Time Fields in GMSEC Messages, 2 of 2	120
Table 5-17. Archive Message Retrieval Request Information Bus Header	121
Table 5-18. Archive Message Retrieval Request Contents	122
Table 5-19. Examples of Start and Stop Times	123
Table 5-20. Archive Message Retrieval Response Summary	124
Table 5-21. Archive Message Retrieval Response Information Bus Header	125
Table 5-22. Archive Message Retrieval Response Contents	126
Table 5-23. Meaning of Response Status and Return Value with Recommended Actions	127
Table 5-24. Archive Message Retrieval Request Subject Naming	128
Table 5-25. Properties of the <i>Miscellaneous Elements</i> for the Archive Message Retrieval Request	128
Table 5-26. Archive Message Retrieval Response Subject Naming	129
Table 5-27. Properties of the <i>Miscellaneous Elements</i> for the Archive Message Retrieval Response ...	129
Table 5-28. Directive Request Message Summary	132
Table 5-29. Directive Request Message Information Bus Header	133
Table 5-30. Directive Request Message Contents	134
Table 5-31. Directive Response Message Summary	135
Table 5-32. Directive Response Information Bus Header	136
Table 5-33. Directive Response Message Contents	136
Table 5-34. Directive Request Message Subject Naming	137
Table 5-35. Properties of the <i>Miscellaneous Elements</i> for the Directive Request Message	137
Table 5-36. Directive Response Message Subject Naming	138
Table 5-37. Properties of the <i>Miscellaneous Elements</i> for the Directive Response Message	138
Table 5-38. Component-To-Component Transfer Message Summary	139

Table 5-39. Component-To-Component Transfer Message Information Bus Header.....	140
Table 5-40. Component-To-Component Transfer Message Subtypes	140
Table 5-41. Component-To-Component Transfer Message Contents for Configuration Status	141
Table 5-42. Group Hierarchical Associations	143
Table 5-43. Component-To-Component Transfer Message Contents for Control	144
Table 5-44. Component-To-Component Transfer Message Contents for Device.....	146
Table 5-45. Component-To-Component Transfer Message Contents for Heartbeat	148
Table 5-46. Component-To-Component Transfer Message Contents for Resource.....	151
Table 5-47. Component-to-Component Transfer Message Subject Naming	153
Table 5-48. Properties of the <i>Miscellaneous Elements</i> for the Component-to-Component Transfer Messages	153
Table 5-49. Telemetry Data Messages and How They Are Used	156
Table 5-50. Telemetry Messages	158
Table 5-51. Telemetry Message Summary.....	158
Table 5-52. Telemetry Message Information Bus Header.....	159
Table 5-53. Telemetry Message Contents for CCSDS Packet.....	160
Table 5-54. Telemetry Message Contents for CCSDS Frame	161
Table 5-55. Telemetry Message Contents for TDM Frame	163
Table 5-56. Telemetry Message Contents for a Processed Telemetry Frame.....	164
Table 5-57. Telemetry Message Subject Naming	167
Table 5-58. Properties of the <i>Miscellaneous Elements</i> for the Telemetry Message	167
Table 5-59. Replay Telemetry Request Message Summary	171
Table 5-60. Replay Telemetry Request Message Information Bus Header.....	171
Table 5-61. Replay Telemetry Request Message Contents	172
Table 5-62. Replay Telemetry Response Message Summary	175
Table 5-63. Replay Telemetry Response Information Bus Header	176
Table 5-64. Replay Telemetry Response Message Contents	176
Table 5-65. Replay Telemetry Request Message Subject Naming.....	177
Table 5-66. Properties of the <i>Miscellaneous Elements</i> for the Replay Telemetry Request Message	177
Table 5-67. Replay Telemetry Response Message Subject Naming	178
Table 5-68. Properties of the <i>Miscellaneous Elements</i> for the Replay Telemetry Response Message...	178
Table 5-69. (Replay) Telemetry Message Subject Naming	179
Table 5-70. Properties of the <i>Miscellaneous Elements</i> for the (Replay) Telemetry Message	179
Table 5-71. Mnemonic Value Request Message Summary	181
Figure 5.3.2.1-1 Mnemonic Value Message Sequence Diagram.....	182
Table 5-72. Mnemonic Value Request Information Bus Header	183
Table 5-73. Mnemonic Value Request Message Contents	184
Table 5-74. Mnemonic Value Response Message Summary	188
Table 5-75. Mnemonic Value Response Information Bus Header	189
Table 5-76. Mnemonic Value Response Message Contents.....	190
Table 5-77. Mnemonic Value Data Message Summary.....	193
Table 5-78. Mnemonic Value Data Message Information Bus Header	194
Table 5-79. Mnemonic Value Data Message Contents	195
Table 5-80. Mnemonic Value Request Message Subject Naming.....	199
Table 5-81. Properties of the <i>Miscellaneous Elements</i> for the Mnemonic Value Request Message	199
Table 5-82. Mnemonic Value Response Message Subject Naming	200
Table 5-83. Properties of the <i>Miscellaneous Elements</i> for the Mnemonic Value Response Message ...	200
Table 5-84. Mnemonic Value Data Message Subject Naming	201
Table 5-85. Properties of the <i>Miscellaneous Elements</i> for the Mnemonic Value Data Message	201
Table 5-86. Archive Mnemonic Value Request Message Summary	204
Table 5-87. Archive Mnemonic Value Request Information Bus Header	205
Table 5-88. Archive Mnemonic Value Request Message Contents	206
Table 5-89. Archive Mnemonic Value Response Message Summary	210

Table 5-90. Archive Mnemonic Value Response Information Bus Header	211
Table 5-91. Archive Mnemonic Value Response Message Contents	212
Table 5-92. Relationship between RESPONSE-STATUS and Dependent Fields	213
Table 5-93. Interpretation of the RESPONSE-STATUS and RETURN-VALUE Fields	214
Figure 5.3.2.2.3-1 Archive Mnemonic Value Message Sequence Diagram	215
Table 5-94. Archive Mnemonic Value Data Message Summary	217
Table 5-95. Archive Mnemonic Value Data Message Information Bus Header	218
Table 5-96. Archive Mnemonic Value Data Message Contents.....	219
Table 5-97. Archive Mnemonic Value Request Message Subject Naming.....	222
Table 5-98. Properties of the <i>Miscellaneous Elements</i> for the Archive Mnemonic Value Request Message	222
Table 5-99. Archive Mnemonic Value Response Message Subject Naming.....	223
Table 5-100. Properties of the <i>Miscellaneous Elements</i> for the Archive Mnemonic Value Response Message.....	223
Table 5-101. Archive Mnemonic Value Data Message Subject Naming	224
Table 5-102. Properties of the <i>Miscellaneous Elements</i> for the Mnemonic Value Data Message	224
Table 5-103. Database Attributes Request Message Summary	226
Table 5-104. Database Attributes Request Information Bus Header	227
Table 5-105. Database Attributes Request Message Contents	228
Table 5-106. Relationship Between ATTRIBUTE-TYPE and Dependent Fields	229
Table 5-107. Database Attributes Response Message Summary	230
Table 5-108. Relationship between RESPONSE-STATUS and Dependent Fields	231
Table 5-109. Database Attributes Response Information Bus Header	231
Table 5-110. Database Attributes Response Message Contents for Limit Sets.....	232
Table 5-111. Database Attributes Response Message Contents for Text Conversion	233
Table 5-112. Database Attributes Response Message Contents for Calibration Curve	234
Table 5-113. Database Attributes Response Message Contents for Short Description	235
Table 5-114. Database Attributes Response Message Contents for Long Description	236
Table 5-115. Database Attributes Response Message Contents for List of All Mnemonics	237
Table 5-116. Meaning of RESPONSE-STATUS and RETURN-VALUE with Recommended Actions	238
Table 5-117. Database Attributes Request Message Subject Naming.....	239
Table 5-118. Properties of the <i>Miscellaneous Elements</i> for the Database Attributes Request Message.....	239
Table 5-119. Database Attributes Response Message Subject Naming	240
Table 5-120. Properties of the <i>Miscellaneous Elements</i> for the Database Attributes Response Message	240
Table 5-121. Command Request Message Summary	242
Table 5-122. Command Request Message Information Bus Header	243
Table 5-123. Command Request Message Contents.....	244
Table 5-124. Command Request Message Summary	245
Table 5-125. Command Response Information Bus Header.....	246
Table 5-126. Command Response Message Contents	247
Table 5-127. Command Request Message Subject Naming	248
Table 5-128. Properties of the <i>Miscellaneous Elements</i> for the Command Request Message	248
Table 5-129. Command Response Message Subject Naming	249
Table 5-130. Properties of the <i>Miscellaneous Elements</i> for the Command Response Message	249
Table 5-131. Product Request Message Information Bus Header	250
Table 5-132. Product Request Message Contents.....	251
Table 5-133. Product Response Information Bus Header	253
Table 5-134. Product Response Message Contents	254
Table 5-135. Meaning of RESPONSE-STATUS and RETURN-VALUE with Recommended Actions	255
Table 5-136. Product Message Information Bus Header.....	256
Table 5-137. Example Scenarios Using the Set of Product Messages	257
Table 5-138. Product Message Contents	258

Table 5-139. Product Request Message Subject Naming	260
Table 5-140. Properties of the <i>Miscellaneous Elements</i> for the Product Request Message	260
Table 5-141. Product Response Message Subject Naming	261
Table 5-142. Properties of the <i>Miscellaneous Elements</i> for the Product Response Message	261
Table 5-143. Product Message Subject Naming	262
Table 5-144. Properties of the <i>Miscellaneous Elements</i> for the Product Message	262
Table 5-145. Product Message Subject Name Template	264
Table 5-146. Simple Service Request Message Summary	266
Table 5-147. Simple Service Request Message Information Bus Header	267
Table 5-148. Simple Service Request Message Contents	268
Table 5-149. Simple Service Response Message Summary	270
Table 5-150. Simple Service Response Information Bus Header	271
Table 5-151. Simple Service Response Message Contents	271
Table 5-152. Simple Service Request Message Subject Naming	273
Table 5-153. Properties of the <i>Miscellaneous Elements</i> for the Simple Service Request Message	273
Table 5-154. Simple Service Response Message Subject Naming	275
Table 5-155. Properties of the <i>Miscellaneous Elements</i> for the Simple Service Response Message	275
Table 5-156. Attitude Parameter Message Contents	280
Table 5-157. Attitude Ephemeris Message Contents	281
Table 5-158. Orbit Parameter Message Contents	284
Table 5-159. Orbital Mean-Elements Message Contents	286
Table 5-160. Orbit Ephemeris Message Contents	287
Table 5-161. Tracking Data Message Contents	290
Table 5-162. Navigation Data Message Subject Naming	291
Table 5-163. Properties of the <i>Miscellaneous Elements</i> for the Navigation Data Message	291
Table 6-1. GMSEC Services (1 of 2)	298
Table 6-2. GMSEC Services (2 of 2)	299
Table 6-3. Sample Mission Services	300
Table 6-4. Flight Dynamics Consumed Services	305
Table 6-5. Flight Dynamics Provided Services	306
Table 6-6. Planning and Scheduling Consumed Services	307
Table 6-7. Planning and Scheduling Provided Services	308
Table 6-8. Archive and Assessment Consumed Services	309
Table 6-9. Archive and Assessment Provided Services	310
Table 6-10. Telemetry and Command Consumed Services	311
Table 6-11. Telemetry and Command Provided Services	312
Table 6-12. Front End Processors and Simulators Consumed Services	313
Table 6-13. Front End Processors and Simulators Provided Services	313
Table 6-14. Automation Consumed Services	314
Table 6-15. Automation Provided Services	314
Table 6-16. Common Consumer Services	315
Table 6-17. Common Provided Services	315
Figure C.1-1. Middleware Overview	332
Table C-1. Publish/Subscribe Scenarios	333
Figure C.2-1. GMSEC Interface Layers	335
Table C-2. API Languages, Platforms, and Middleware	338

Section 1 Introduction

1.1 Background

The GMSEC mission is to provide a flexible and cost-effective means to meet the operational needs of current and future missions. This includes future constellation missions consisting of multiple spacecraft performing identical observations or collaborative observations. In order to provide rapid and flexible mission development and continued operations throughout a mission's lifecycle, the GMSEC architecture incorporates a concept that easily integrates, adds and deletes ground and flight system functional components. The GMSEC architecture is a middleware-based system architecture using standardized messages, an application programming interface, and utilizing COTS and GOTS components. Initially designed for NASA/GSFC missions, its application and deployment have extended beyond this arena.

Features for this concept include a software component "plug and configure" capability that will insure less development and integration time, and lower costs to upgrade evolving government-off-the-shelf (GOTS) and commercial-off-the-shelf (COTS) components. It also promotes platform independence that will allow easy intercommunication between components across different operating systems (OS). The *GMSEC Architecture Document* (see Section 1.4.1 GMSEC Documents) provides a detailed description of the concepts, high level design, and application of this architecture.

1.2 Overview

GMSEC has defined a set of standard messages to facilitate communication with other sub-systems and components. It also has implemented an Applications Programming Interface (API) for the various components to have a uniform interface to the underlying middleware (and isolate the component from the middleware). Using the standard messages along with the GMSEC API will allow a component to be GMSEC compliant and help it achieve plug and configure compatibility. For a component to be considered GMSEC compliant, it must use the standard set of messages with the GMSEC API.

The GMSEC Interface Specification Document contains the standard set of defined messages. Each GMSEC standard message contains a GMSEC Information Bus Header section and a Message Contents section. Each message section identifies required fields, optional fields, data type and recommended use of the fields. Additionally, this document includes the message subjects associated with the standard messages. The system design of the operations center should ensure the components that are selected use both the API and the defined standard messages in order to achieve full interoperability from component to component. The messages are described in detail in Section 5 GMSEC Standard Messages.

The GMSEC Application Programming Interface (API) was defined and implemented to aid in packing, sending, receiving, and unpacking GMSEC standard messages. The API provides a standard interface to the various middleware that are responsible for the routing and delivery of messages. The use of the GMSEC API will also allow compliant middleware to be swapped in and out without requiring application software modifications. Section 6 GMSEC Application Programming Interface provides an overview of the GMSEC API. Greater detail on the API is available in its associated documentation. See Section 1.4.1 GMSEC Documents.

1.3 Scope of the Interface Specification

The following sections describe areas relevant to the GMSEC Architecture but fall just outside the scope of the Interface Specification Document.

1.3.1 File Transfers

The GMSEC architecture has developed an interface and mechanism to transport messages between components. Or, more accurately components interact with the GMSEC Information Bus to publish and receive messages to affect **component-to-component** message transfers. The GMSEC architecture has not directly addressed **location-to-location** file transfers, that is, moving a file or product from one physical location to another (or even a one-to-many locations distribution). The GMSEC architecture and implementation allows for products and files to be incorporated into its message structure to then be sent on the GMSEC Information Bus, **component-to-component**. The architecture has left to the components and missions to determine how (to push or pull) or where (to an archive, library, or storage facility) such a product or file should be stored as well as determining access, authorization, and authentication to the storage locations.

There may be a need for a separate component (or service) to be available on the GMSEC Information Bus to move files around. This component/service could be invoked by other components wanting to “hire out” the file & product movement responsibilities to an expert. A separate file/product transfer component would accept requests by other components to move these objects from one location to another, with one-to-one or one-to-many distribution capabilities. This component might also interact with a partnering component to move files across operating systems, networks, sites, gateways, or portals. This component would also take on security, access & authorization concerns, and attempt guaranteed product delivery relieving other components of these responsibilities.

1.3.2 GMSEC Compliant Components

Please see *Section 3.7 GMSEC Interoperability* in the GMSEC Architecture Document (Release 2.8, September 2013) for a discussion on the term “GMSEC compliant”.

1.4 Applicable and Referenced Documents

1.4.1 GMSEC Documents

The following GMSEC documents set forth the GMSEC Architecture, the GMSEC message specifications, and the GMSEC Applications Programming Interface. Documents for specific GMSEC-compliant software components should be consulted on an individual basis.

- GMSEC Architecture Document, Release 2.8, September 2013
- GMSEC Interface Specification Document, 2016 March (this document)
- GMSEC API 4.0 User’s Guide, March 2016

1.4.2 Existing Space Data Systems Standards

1.4.2.1 Telemetry and Command Data References

In relation to telemetry data formats, particularly the Consultative Committee for Space Data Systems (CCSDS) frames and packets, the CCSDS Recommendations and Reports should be referenced and can be found at the following web site:

<http://www.ccsds.org>

1.4.2.2 XML Telemetric and Command Exchange (XTCE) Reference

The XML Telemetric and Command Exchange (XTCE) data specification provides an information model for telemetry and command data. This specification defines a standard exchange format for telemetry and commanding that will support the exchange of data through all phases of the satellite, payload, and ground segment lifecycle: system design, development, test, validation, and mission operations. It is found at the following web site:

<http://www.omg.org/space/xtce/>

1.4.3 Potential Future Space Data Systems Standards

1.4.3.1 Ground Operations Automation Language

Ground station operations use manual and scripted procedures to command and configure ground equipment; reference, process and analyze data; and command the spacecraft, instrument, or payload. Different ground systems have implemented different languages for their own operations that are often incompatible yet similar in functionality. A common Ground Operations Automation Language would make transitions and upgrades easier and less expensive, allow consistency along the whole life cycle of development, test and operations, and make automation much more viable and trustworthy. Developing a standard specification to access, monitor, and control these assets will simplify operations, reduce development costs, increase component sharing, and facilitate the plug and configure environment.

In 2004, the OMG sought input for a “Ground Operations Automation Language” by issuing a Request for Information. In 2009 a “Specification for the Spacecraft Operations Language Metamodel” was released that included definitions and requirements for a proposed meta-model, stating: “A standard meta-model to represent spacecraft operations procedures will facilitate the transfer of procedures between the spacecraft vendor and the spacecraft operator, as well as allow for maintenance and transfer of the procedures across different ground systems employed over the lifetime of the spacecraft.” The Spacecraft Operations Language Metamodel would enable the definition of a platform independent model of a spacecraft procedure.

1.4.3.2 Spacecraft Monitor and Control

The Consultative Committee for Space Data Systems (CCSDS) in the area of Mission Operations and Information Management Systems (MOIMS) has a Spacecraft Monitor and Control (SM&C) Working Group. SM&C refers to end-to-end services between onboard applications and ground-based functions responsible for mission operations. The scope of this group includes the definition of a set of concepts, reference architecture, and a service framework for spacecraft monitoring and control. A number of documents have been produced by the group that are applicable now and in the future in the evolution of the GMSEC architecture. These documents include:

- *Mission Operations Services Concept*
- *Mission Operations – Message Abstraction Layer*
- *Mission Operations – Reference Model*
- *Mission Operations – Common Object Model*
- *Spacecraft Monitor and Control – Common Services*
- *Spacecraft Monitor and Control – Core Services*
- *Asynchronous Message Service*

See their web site at:

<http://www.ccsds.org>

1.4.4 Document Creation Tools

UML Sequence Diagrams found in this document were generated using the “WebSequenceDiagrams” tool found at:

<http://www.websequencediagrams.com/>

Section 2 Quality of Communications Service

Within the GMSEC architecture, exchangeable and diverse component-based applications communicate with one another using a middleware infrastructure. The software applications are connected to the middleware, which is then responsible for message routing. The middleware acts as a software information bus or software backplane service to transfer messages between applications. The applications are freed from having to know about the existence of other components, their location, or from managing the details of the communication link.

The software information bus will provide a number of communication mechanisms to the software components. The two primary mechanisms are Publish/Subscribe and Session-based communications. Briefly described:

- **Publish/Subscribe** – each component publishes its messages to the software information bus for all other components to receive, and subscribes to the information bus for all messages and data that it requires.
- **Session-based** – Two cooperating components find and interact with each other directly. This communication is also known as Request/Reply.

Request/Reply communications, where the sender/requester of information can receive back information from the provider, is typically associated with point-to-point communications. Publish/Subscribe communications, where the sender/publisher may not receive any information back from the subscriber, is typically associated with multicast communications. The following table compares these two communication models.

Table 2-1. Communication Model Comparison

Communication Model	Typical Transport Mechanism	Driven By:	Interaction	Application Coupling
Publish/Subscribe where data producers publish data to the network at large	Multicast, or message replication	Events, such as the arrival of data	One-to-Many, with one multicast message published once, received by all subscribers	Very loose; by message subject (or topic) only
Request/Reply where data consumers coordinate closely with data producers	Point-to-Point	Demand for data or service	One-to-One, with a client requesting data, and the server returning an individual response	Very close; on a message-by-message basis.

With each communication mechanism there is an associated quality of service. That is, how reliable is the delivery of messages? And, how is the quality of service described? Descriptions of the kinds of delivery service follow.

Reliable Delivery

- Is attempted over a specified time range, or until the restart of either the application software or the underlying communications hardware/software
- Will overcome short network interruptions, but not an application “outage”, that is, stopping and re-starting of an application
- Normally, has no advisory messages indicating that delivery was successful
- Is typically associated with multicast delivery

Guaranteed Delivery

- Will assure delivery of every message within the time constraints specified for each message
- Advisory status messages will be sent, including in the event delivery is not achieved
- Can be achieved by the applications and/or the underlying COTS software
- Is typically associated with point-to-point links

Persistence – a special case of Guaranteed Delivery

- Is available only with Guaranteed Delivery
- Will attempt delivery beyond (despite) the restart of the application process or the underlying communications hardware/software
 - For GMSEC, newly connected applications will receive the last message of a requested message type
 - What capability the underlying middleware provides will be made available to the application by the GMSEC API
- Is typically used with point-to-point communications, but can be used in a one-to-many distribution

Combining Communication Models and Transport Mechanisms

In addition to the types and quality of services listed above, applications can also combine the communication delivery and transport mechanisms to achieve their desired communication model and quality of service. For example:

1. Multicast communication, Reliable but not Guaranteed, can be combined with point-to-point communication.

- a. The Reliable application publisher could establish a message exchange protocol where it requires the subscribers to respond back to the publisher with their own published message to simply confirm receipt. This will effectively create a form of guaranteed communication implemented at the application level.
 - b. An application may publish (multicast) a message. The receiving application(s) may respond to a certain subject or message type by sending a point-to-point informational message back to the publisher. This is effectively a Request/Reply exchange using a Publish/Subscribe mechanism.
2. Request/Reply, normally associated with point-to-point communication, can be combined with multicast.
- a. An application might send a request to another application requesting that a message (or data) commence publishing. The responding application would send a Reply and include the message subject under which the data will be published. The requesting application would subscribe to the published messages.
 - b. An application might have a need for a number of point-to-point links to a number of applications. It would send the same request once over each communication link, to achieve guaranteed message delivery by means of a “send-multiple-times-cast” rather than a true single multicast.

Thus, the application components have a good deal of flexibility and mechanisms to implement the type and quality of the needed communication service. Each application can determine what level of service is required and implement that service with a single or combination of mechanisms. Of course, when possible, the applications should use the available mechanisms and quality of service that are provided by the underlying middleware. For example, if guaranteed delivery or data persistence is needed, use what is provided by the middleware through the services of the API.

The following table summarizes the aspects of the various delivery mechanisms.

Table 2-2. Speed and Quality of Delivery Mechanisms

Delivery Mechanism	Delivery Quality	Throughput Performance
Reliable	Good	Fastest
Guaranteed	Excellent	Good
Guaranteed with Persistence	Best	Slowest

Performance

Every system must be properly designed and engineered to handle the nominal and worst case throughput scenarios. If not, performance related problems could be expected.

It is possible for some messages (Log, Telemetry, Mnemonic Value Data, and Product Messages with large embedded files) to flood or surpass the capacity of the underlying computers, network equipment, or processing software. Or, to be more accurate, it is not the messages but the components sending the volume of data that could exceed the capacity either in short bursts or sustained traffic.

While the middleware, API, and message definitions will add a small overhead to the volume of data to be transferred, they will not significantly deplete the resources of the network bandwidth or the processing cycles of the computers (as demonstrated with performance tests). It is the volume of data itself (sans the software information bus) that must be addressed, and for which the entire system must be appropriately designed and engineered.

On occasion, the components themselves may meter, throttle, or “thin” the number of messages so as to not exceed the throughput capacities of the underlying equipment. But, the best approach is to build in sufficient capacity (in network bandwidth and computer resources) to handle the sustained traffic load as well as the expected short burst.

The GMSEC API will not itself perform any monitoring or self-throttling of data to reduce the throughput. A properly engineered system will, from the outset and when significant changes in requirements occur, make account for sufficient network bandwidth, computing resources, middleware, operating systems, and software components for the throughput performance required.

Section 3 GMSEC Message Subject Definition

Subject names are constructed to provide the subscribing applications efficient flexibility to filter messages. Filtering will occur at two levels, at the middleware level and at the application level. Filtering will take place at the middleware level based solely upon the subject name. The middleware will not extract information from the message contents as part of its filtering since it may not have access to message formats because both GMSEC standard and non-GMSEC messages will be transported. Filtering at the application level will take place based upon the message contents. Applications should be built and subjects defined so as to maximize filtering at the subject level by the middleware. Little to no filtering of messages should occur at the application level since this will result in wasted use of resources by the middleware to provide an application with a message it will immediately discard. The following table suggests some of the items upon which the subscriber would filter a message. Not every message or possibility is listed.

Table 3-1. Sample Subject Filtering Items in Addition to Type and Subtype

Message Description	Message Type	Message Subtype Name	Other Subject Filtering Items
Real-Time Log Message	MSG	LOG	<ul style="list-style-type: none"> • Mission/Constellation • Satellite • Publisher • Occurrence type • Severity
Archive Message Retrieval Request and Response	REQ, RESP	AMSG	<ul style="list-style-type: none"> • Time • Any field in the log message
Directive Request and Response	REQ, RESP	DIR	<ul style="list-style-type: none"> • Requestor • Response Status (ACK, Working, Success, Failure)
Telemetry Messages	MSG	TLM	<ul style="list-style-type: none"> • Mission/Constellation • Satellite • Publisher • Telemetry format • Stream mode • Channel or AP ID
Replay Telemetry Message	REQ, RESP	RTLTM	<ul style="list-style-type: none"> • Same as Telemetry Message
Mnemonic Value Data Message, Request, and Response	MSG, REQ, RESP	MVAL	<ul style="list-style-type: none"> • Mnemonic • Requestor
Archive Mnemonic Value Data Message, Request, and Response	MSG, REQ, RESP	AMVAL	<ul style="list-style-type: none"> • Mission/Constellation • Satellite • Requestor
Database Attributes Request, and Response	REQ, RESP	DB	<ul style="list-style-type: none"> • Mission/Constellation • Satellite • Requestor
Product, Product Request, and Product Response	MSG, REQ, RESP	PROD	<ul style="list-style-type: none"> • Mission/Constellation • Satellite • Publisher • Product Type and Subtype

3.1 Characteristics of GMSEC Message Subject Names

1. The set of GMSEC messages must be easily distinguished in the subject with short elements encouraged for fast parsing and efficient throughput.

2. In order for the middleware to distinguish standard GMSEC messages from other routable messages, an indicator or element should be present in the subject name. Thus, GMSEC standardized messages can be filtered and routed apart from other non-GMSEC messages.
3. The GMSEC element portion of the subject will have a naming convention that can easily identify the message type (and subtype)
 - Such as, request, reply, unsolicited (self-initiated such as a log message), data stream
4. Most messages will want to be filtered and thereby subscribed to by a common set of parameters, but a common set of parameters may not be applicable to all message subjects. Therefore, applications will use the common set of parameters, but also be able to expand upon these common filtering parameters with their own unique filtering parameters.
 - Examples of these include constellation and/or satellite, message type and subtype; therefore, a subject name should contain elements with these common distinguishing characteristics.
5. A means of distinguishing telemetry data streams (or other data/messages streams) is desired at the subject name level so that an application can subscribe to a single or even multiple sets of data/message streams.
 - An application may want to subscribe to
 - A future telemetry data stream or a subset of future data streams with a single telemetry format or a subset of telemetry formats, from a single satellite or a constellation of satellites
 - Real-time or playback data streams
 - A subset of mnemonics from a single satellite
 - A configuration table of active or future (expected) data streams could assign a unique Stream ID to each data stream that the publisher would then include in the subject name. A subscriber could look up the Stream ID in the table and subscribe to the subject that includes that unique ID. Furthermore, the publisher of the data may (request to) update the data stream table with the subject by which the data will be published. (This could also be used to distinguish a real-time stream from a playback stream. Subscribers must also determine if they need to unsubscribe once the data stream has ceased.)

3.2 Format of GMSEC Message Subjects (or Topics)

Based on the characteristics of the GMSEC Message Subject Names and on existing subject (topic) conventions, the following subject name format is being used.

- Message subject names will follow the format of a string of characters separated by the dot (.) character. The character strings separated by the dots are called elements.

- THIS.IS.A.VALID.SUBJECT (This subject has 5 elements)
- Only UPPERCASE alphanumeric characters, the underscore “_” character, and the “-” dash character are to be used. No invisible or control type characters shall be used. No element shall be empty. If an element is required but not applicable to that type of message, the publisher shall insert “FILL” (no quotes) for that element. For example, most, but not all messages are satellite related, so a “Satellite ID” would be part of all message subjects. If a message were not satellite specific, the publisher would insert “FILL” into that portion of the subject name.
 - THIS.IS.NOT.A.VALID.SUBJECT. (missing element at the end)
 - .THIS.IS.NOT.A.VALID.SUBJECT (empty element at beginning)
 - THIS.IS..NOT.A.VALID.SUBJECT (empty element in middle)
 - GMSEC.FILL.VALID.SUBJECT (valid GMSEC subject)
- A GMSEC subject will be distinguished from other subjects by the first element, which shall contain the capitalized text “GMSEC” (no quotes). For non-GMSEC subjects (or messages) created by missions or vendors, the subject will be case sensitive, that is, UPPERCASE elements are not enforced.
 - GMSEC.VALID.SUBJECT (valid GMSEC subject)
 - Gmsec.not.valid.subject (invalid GMSEC subject, lower case)
- An asterisk (*) can take the place of one whole element (not a substring of an element) and act as a wildcard. An application can subscribe to subjects using wildcards. A publisher of a message CANNOT use the wildcard character in that it could produce unwanted and wildly variable effects. If an element is required but not applicable to the subject, the publisher MUST insert “FILL” into that element.
 - THIS.*.VALID (valid wildcard substitution)
 - THIS.IS*.NOT.VALID (invalid wildcard substitution)

- A Greater-than (>) character can appear ONLY as the right-most character of a subject immediately after the element delimiter (".") and will match all elements to the right.
 - THIS.IS.VALID.> (valid wildcard substitution)
 - THIS.IS.>.NOT.VALID (not a valid use of "match all to the right")
- A Plus (+) character can appear ONLY as the right-most character of a subject immediately after the element delimiter (".") and will match all elements to the right, but there is no requirement for additional elements. For example, the example "THIS.IS.VALID.+" will match the subjects "THIS.IS.VALID" and "THIS.IS.VALID.SUBJECT".
 - THIS.IS.VALID.+ (valid wildcard substitution)
 - THIS.IS.+.NOT.VALID (not a valid use of "match all to the right")

The following examples illustrate the use of the "*" and ">" wildcard syntax and the matching semantics. Since GMSEC subject definitions require uppercase characters, case sensitivity is not an issue.

- THIS.IS.VALID.* Match the first 3 elements and any fourth element. Here, subjects with more than 4 elements will not match.
- THIS.IS.VALID.> As long as the first three elements match, will match any subject of any greater length. Subjects of three elements or less will not match.

Table 3-2. Subject Matching Examples

Subject	Matching Subjects	Non-Matching Subjects	Why
ONE.TWO.*	ONE.TWO.THREE	ONE.TWO.THREE.FOUR	Extra element
	ONE.TWO.SEVEN	ONE.TWO	Missing element
	ONE.TWO.TWO	ONE.TWOTHREE.FOUR	Non-matching second element
ONE.>	ONE.TWO	TWO.ONE	Position mismatch
	ONE.TWO.THREE	ONE	Missing element
	ONE.TWO.XYZ.FIVE	ONEZ.TWO	Non-matching first element
ONE.+	ONE	TWO.ONE	Position mismatch
	ONE.TWO	ONEZ.TWO	Non-matching first element
	ONE.TWO.XYZ.FIVE		

- In order to maximize speed and throughput rates, subject names should be short and not use an extraordinary number of elements.
 - The length of an element should attempt to not exceed 12 characters.
 - The length of a subject is dependent on the number of elements.

-

3.3 Standards of GMSEC Message Subject Names

A few common elements of the subject can be identified that would (nearly) always be included in the subject. They are:

- Subject definition standard
- Mission and/or constellation ID
- Satellite ID
- Message type
- Message subtype

Additionally, application programs should be able to define their own set of unique elements of the subject in order to create their own unique subject names. Therefore, a GMSEC-defined message subject will contain a fixed portion and a variable portion of elements, and defined to look like the following:

Table 3-3. GMSEC Message Subject Name Definition

Subject Element	Subject Standard	Mission Elements		Message Elements		Miscellaneous Elements			
	Specification	Mission	Sat ID	Type	Subtype	ME1	ME2	ME3	ME4...
	FIXED PORTION					VARIABLE PORTION			
	Required Elements					Message definition determines whether a Miscellaneous Element is required or optional			

In textual format the subject would appear as:

SPECIFICATION.MISSION.SATID.TYPE.SUBTYPE.ME1.ME2.ME3... AND SO ON.

GMSEC fixes the first 5 subject elements, the **Specification**, **Mission (2)**, and **Message (2)** elements. These elements are always defined the same and are required to be filled in by the publisher/sender of a message.

The elements to the right of the fixed portion of the GMSEC subject are the **Miscellaneous Elements** and variable portion of the subject. These elements are message and subscriber specific. That is, the publisher/subscriber (sender/receiver) would predefine or even dynamically create as many *miscellaneous elements* as needed (a variable number of) according to their filtering needs. Thus, depending on the message definition, they can have different meanings, can vary in number, and be either required or optional.

If an element is required but not applicable to that type of message, the publisher shall insert "FILL" (no quotes) for that element. For example, most, but not all messages are satellite related, so a "Satellite ID" would be part of all message subjects. If a message were not satellite specific, the publisher would insert "FILL" into that portion of the subject name.

3.3.1 Subject Standard Element of the GMSEC Message Subject

Table 3-4. Subject Standard Element of the GMSEC Message Subject

	Subject Standard	Mission Elements		Message Elements	
Subject Element	Specification	Mission	Sat ID	Type	Subtype
	FIXED PORTION				

The Subject Standard element of the message subject identifies the specification used for the message subject. In this instance, this interface specification document is the standard by which the GMSEC message subject is defined and interpreted. For GMSEC defined message subjects, the first element is always “GMSEC”. Other message definition standards and their associated message subjects can be used within the GMSEC architecture with the GMSEC API. However, it must be noted that the GMSEC API inserts data into some GMSEC message fields prior to publishing. (See Section 4.1.1.2 Required, Optional, and API Field Indicator for a discussion on these fields.) These automatic insertions into the “tracking fields” by the API must be disabled when using non-GMSEC message definitions. See the *GMSEC API Users Guide* for further information.

3.3.2 Mission Elements of the GMSEC Message Subject

Table 3-5. Mission Elements of the GMSEC Message Subject

	Subject Standard	Mission Elements		Message Elements	
Subject Element	Specification	Mission	Sat ID	Type	Subtype
	FIXED PORTION				

Two elements comprise the fixed Mission Elements. They are described in the table below.

Table 3-6. Descriptions of the Mission Elements of the GMSEC Message Subject

Element Name	Value	Description
Mission	[Name of mission]	Name of a mission or constellation. E.g., MY-MISSION (in UPPERCASE)
Sat ID	[Name of satellite]	Name of a satellite for that mission or within the constellation of satellites. E.g., MY-PRIME, MY-CONST1, MY-SAT1, ... (in UPPERCASE)

For single satellite missions, the mission identifier and satellite ID may be different, or may be identical. Or, the mission may choose to name the satellite by adding “1” to the mission name. For example,

GMSEC.SUN.HELIO.MSG.LOG... or,
GMSEC.SUN.SUN.MSG.LOG... or,
GMSEC.SUN.SUN1.MSG.LOG...

Some missions may consist of a fleet or a constellation of satellites and can be distinguished in the following manner similar to a product manufacturers model and serial number identification:

GMSEC.MOON.LUNAR1
GMSEC.MOON.LUNAR2
GMSEC.MOON.LUNAR3

GMSEC.MARS.11
GMSEC.MARS.12
GMSEC.MARS.13

Some missions may even have multiple constellations of satellites. If so, the mission ID can take on the constellation identifier (series or model) as in the following examples:

GMSEC.CONST-A.SAT1	GMSEC.CONST-B.SAT1
GMSEC.CONST-A.SAT2	GMSEC.CONST-B.SAT2
GMSEC.CONST-A.SAT3	GMSEC.CONST-B.SAT3

It is important to note that subject naming convention for the “Sat ID” element of the subject does not have to refer to a single physical satellite, though that may be a common way of using the “Sat ID” element. “Sat ID” could refer to the physical satellite (perhaps by flight model number), to a logical satellite name such as “CONTROLLER”, “PRIME”, “EAST”, “RING-A1”, “SPARE2”, or even to a group of satellites such as “RING-A”, “EQUATORIAL-SET”, and “TETHERED”.

Creative use of the “Mission” and “Sat ID” elements to refer to a physical, logical, group (subset), or entire constellation of satellites is possible and permits great latitude for categorization and unique identification of assets.

In generic terms, the mission elements are simply the taxonomy of classifying groups (or sets) and group members (elements or objects).

3.3.3 Message Elements of the GMSEC Message Subject

Table 3-7. Message Elements of the GMSEC Message Subject

	Subject Standard	Mission Elements		Message Elements	
Subject Element	Specification	Mission	Sat ID	Type	Subtype
	FIXED PORTION				

Two elements comprise the fixed **Message** elements. The **Type** element is used to describe the kind of message communication used within GMSEC. These are the **Request** message, the **Response** (or reply) message, and the (basic) **Message**. The **Message** is published without a required or expected response (though it may cause an action when received). The **Request** message is used to request a specific action or information from a service or data provider, or product generator. The **Request** message may or may not require a reply message. If so, the **Response** message is used. Message types are discussed in detail in Section 4.2 GMSEC Messages: Their Characteristics and Interactions. The **Subtype** element contains the ID or name of the message definition. The **Message** elements are listed in the table below.

Table 3-8. Descriptions of the Message Elements of the GMSEC Message Subject

Element Name	Value	Description
Type Kind or intention of communication	MSG	Message
	REQ	Request
	RESP	Response
Subtype Name or ID of GMSEC Defined Message	AMSG	Archive Message Retrieval
	AMVAL	Archive Mnemonic Value Retrieval
	C2CX	Component-to-Component Transfer
	CMD	Command
	DB	Database Attributes
	DIR	Directive
	LOG	Log (or event)
	NDM	Navigation Data Message
	MVAL	Mnemonic Value
	PROD	Product
	RTLM	Replay Telemetry
	SERV	Service
	TLM	Telemetry

3.3.4 Miscellaneous Elements of the GMSEC Message Subject

Table 3-9. *Miscellaneous Elements of the GMSEC Message Subject*

Subject Element	Subject Standard	Mission Elements		Message Elements		Miscellaneous Elements			
	Specifica- tion	Mission	Sat ID	Type	Subtype	ME1	ME2	ME3	ME4...
	FIXED PORTION					VARIABLE PORTION			
	Required Elements					Message definition determines whether a Miscellaneous element is required or optional			

The **Miscellaneous** (and variable) elements are dependent upon a number of factors as outlined below. The first factor is the TYPE of the message.

Table 3-10. Message Type Determines Content of the *Miscellaneous Elements*

Message Type	Meaning	Miscellaneous Elements			
		ME1	ME2	ME3	ME4...
REQ	Request	Responder	Undefined		
RESP	Response	Requestor	Status	Undefined	Undefined
MSG	Message	Publisher	Message specific		

If TYPE = REQ, then **ME1** = component (or group or service) name of responder
ME2, ME3, ... undefined

If TYPE = RESP then **ME1** = component (or group) name of requestor
ME2 = status of the request
ME3... ME5, ... undefined

If TYPE = MSG, then publisher
ME1 = component (or group) name of
ME2, ME3, ... are message specific

The other primary factor that determines the value of the **Miscellaneous elements** is the Message Subtype (or abbreviated name of the message). The **Miscellaneous elements** are further defined in Section 5 GMSEC Standard Messages where the specific GMSEC messages are delineated.

For the *ME1* variable element, group names can also apply. Components may organize themselves into logical associations or groups (pre-defined or dynamically formed) for more creative forms of communication. For example, any one member of a group may need to send or receive a message to/from all members of the group. Using group names will make message subscribing much simpler. This concept is nearly identical to that described in Section 3.3.2 Mission Elements of the GMSEC Message Subject where the “Sat ID” element could be used to refer to a physical, logical, group, or constellation of satellites.

The *ME1* element can also be used as the name of a service for the Simple Service Request and Response Messages. See Section 5.4.2 Simple Service Messages for further information.

3.3.3.1 Service Disposition

Service Providers (software components) may be required to service a high volume of requests, or requests that require a large amount of resources. Furthermore, these high volume or high resource requests should not impede other less demanding requests. A first-in, first-out queue may not be appropriate for all circumstances. If possible, requests that require extended time or resources should be offloaded where they can be more efficiently processed and/or not impact less demanding requests. Thereby, low demand requests could be processed in a timely manner and be completed without being delayed by more resource-heavy requests.

One method to handle this kind of influx is to create and manage a bank (group) of sub-components to service the requests. A service “disposition-er” may have at its disposal a number (1 to n) of readily available service “handlers” to process the requests. The service dispositioner is known by a generic service name to which the service consumers can send requests. Service consumers need only direct their requests to the generic service name (the service provider and dispositioner) who will in turn disposition the requests to its bank of service handlers.

If request volume, size, or frequency is not a problem, a single component could serve in both roles of a service dispositioner and a service handler. In this case, the component name, service name, service dispositioner name, and service handler are all the same term and there is no distinction of roles.

For an example of the case of a dispositioner and handler, a telemetry and command service provider may be known as TAC. It may have a number of associated service handlers named TAC1, TAC2, TAC3, etc. The service

dispositioner subscribes to messages directed to the generic service provider TAC. It delegates, or dispositions the requests to one of its group of service providers: TAC1, TAC2, or TAC3. The service dispositioner can, with the aid of the underlying middleware capabilities, also implement round-robin distribution, load balancing, or other techniques (though this may lock in to a specific vendor). It is the responsibility of the service dispositioner to develop and define the mechanisms of the interaction with its service handlers.

At this time, GMSEC does not provide any specific mechanism for this capability. It is left up to the mission design and implementation to utilize the flexibility of the GMSEC message subjects.

3.4 Definitions of GMSEC Subject Names

Table 3-11. Message Subject Syntax

	Subject Standard	Mission Elements		Message Elements		Miscellaneous Elements			
Subject Element	Specification	Mission	Sat ID	Type	Subtype	ME1	ME2	ME3	ME4...

SPECIFICATION.MISSION.SATID.TYPE.SUBTYPE.ME1.ME2.ME3.ME4.ME5...

The following table is a list of defined GMSEC message subjects. The first three fixed elements (Subject Standard and two Mission Elements) are not shown. As stated in the previous sections, the *ME1* element could take the form of a targeted component, service, group, or other subscribe-able term. *ME6* and greater are not shown.

Table 3-12. GMSEC Message Subject Definitions, 1 of 4

Message Name	Subject Elements						
	Message Elements		Miscellaneous Elements				
	TYP	SUBTYP	ME1	ME2	ME3	ME4	ME5...
Control and Monitor Level							
Archive Message Retrieval Request	REQ	AMSG	Service Provider				
Archive Message Retrieval Response	RESP	AMSG	Service Consumer	Status			
Component-to-Component Transfer	MSG	C2CX	Publisher	C2CX Subtype: [CFG, CNTL, DEV, HB, RSRC]	Intended recipient (destination) component (optional)		
					Component	Node	Facility
Directive Request	REQ	DIR	Service Provider	[DIRECTIVE-KEYWORD]			
Directive Response	RESP	DIR	Service Consumer	Status			
Log	MSG	LOG	Publisher	[Subclass: ARC, CFG, CMD, ... DIR, ... TLM]	[Occurrence: AOS, LOS, ...RED, YEL]	[Severity: 1-routine, 2-med,...,4-critical]	m5 = [user] (optional), m6 = [refID] (optional)

Table 3-13. GMSEC Message Subject Definitions, 2 of 4

Message Name	Subject Elements						
	Message Elements		Miscellaneous Elements				
	TYP	SUBTYP	ME1	ME2	ME3	ME4	ME5...
Data Level							
Archive Mnemonic Value Data Message	MSG	AMVAL	Publisher				
Archive Mnemonic Value Request	REQ	AMVAL	Service Provider				
Archive Mnemonic Value Response	RESP	AMVAL	Service Consumer	Status			
Command Request	REQ	CMD	Service Provider				
Command Response	RESP	CMD	Service Consumer	Status			
Database Attributes Request	REQ	DB	Service Provider				
Database Attributes Response	RESP	DB	Service Consumer	Status			
Mnemonic Value Data Message	MSG	MVAL	Publisher	Status			
Mnemonic Value Request	REQ	MVAL	Service Provider				
Mnemonic Value Response	RESP	MVAL	Service Consumer	Status			
Telemetry Message	MSG	TLM	Publisher	Stream-Mode	Format-Type	Virtual channel ID	AP ID
Replay Telemetry Request	REQ	RTLTM	Service Provider				
Replay Telemetry Response	RESP	RTLTM	Service Consumer	Status			

Table 3-14. GMSEC Message Subject Definitions, 3 of 4

Message Name	Subject Elements						
	Message Elements		Miscellaneous Elements				
	TYP	SUBTYP	ME1	ME2	ME3	ME4	ME5...
Products & Services Level							
Product Message	MSG	PROD	Publisher	Product Type	Product Subtype 1	Product Subtype 2	Product Subtype 3
Product Request	REQ	PROD	Service Provider				
Product Response	RESP	PROD	Service Consumer	Status			
Simple Service Request	REQ	SERV	Service Name or Service Provider	[Service Group]	[Service Operation]		
Simple Service Response	RESP	SERV	Service Consumer	Status			

Table 3-15. GMSEC Message Subject Definitions, 4 of 4

Message Name	Subject Elements						
	Message Elements		Miscellaneous Elements				
	TYP	SUBTYP	ME1	ME2	ME3	ME4	ME5...
Navigation Data Messages							
Attitude Parameter	MSG	NDM	Publisher	Stream-Mode	Nav-Type		
Attitude Ephemeris	MSG	NDM	Publisher	Stream-Mode	Nav-Type		
Orbit Parameter	MSG	NDM	Publisher	Stream-Mode	Nav-Type		
Orbit Mean-Elements	MSG	NDM	Publisher	Stream-Mode	Nav-Type		
Orbit Ephemeris	MSG	NDM	Publisher	Stream-Mode	Nav-Type		
Tracking Data	MSG	NDM	Publisher	Stream-Mode	Nav-Type		

3.5 Example of the GMSEC Message Subject

To show how to use the GMSEC message subject, the GMSEC DIR Message will be used as an example. The DIR Message is used for Request messages and Response messages. This message exchange could begin with a Directive Request message being sent to a cooperating application. The receiving application responds with a Directive Response message. Below is the directive subject definition.

Table 3-16. Directive Request Message Subject Naming

	Subject Standard	Mission Elements		Message Elements		Miscellaneous Elements		
Subject Element	Specification	MISSION	SAT	TYP	SUBTYP	ME1	ME2	ME3
Subject Content	GMSEC	[mission]	[sat]	[REQ]	DIR	[Component: APP1, TLM2, TLM3 ...]		
Example for Publisher / Sender	GMSEC	MSSN	SAT1	REQ	DIR	APP1		
Example for Publisher / Sender	GMSEC	MSSN	SAT1	REQ	DIR	TLM2		
Example for Subscriber / Receiver	GMSEC	MSSN	SAT1	REQ	*	TLM3		

Table 3-17. Properties of the *Miscellaneous Elements* for the Directive Request Message

<i>Miscellaneous Element</i>	Required / Optional	Description	Field in Msg, if applicable
ME1	Required	Component name of Responder	NA
ME2	Not used		
ME3	Not used		

Examples:

Two components, APP1 and TLM2, interact with the Directive Request Message.

APP1 subject to send the Directive Request to TLM2:

GMSEC.MSSN.SAT1.REQ.DIR.TLM2

TLM2 subject to receive Directive Request Messages targeted to it for the SAT1 satellite:

GMSEC.MSSN.SAT1.REQ.DIR.TLM2

TLM2 subject to receive any Directive Request message targeted for TLM2:
GMSEC..REQ.DIR.TLM2**

TLM2 subject to receive any Request Message targeted for TLM2:
GMSEC..REQ.*.TLM2**

Table 3-18. Directive Response Message Subject Naming

	Subject Standard	Mission Elements		Message Elements		<i>Miscellaneous Elements</i>		
Subject Element	Specification	MISSION	SAT	TYP	SUBTYP	ME1	ME2	ME3
Subject Content	GMSEC	[mission]	[sat]	[RESP]	DIR	[Component: APP1, TLM2, TLM3 ...]	[Status]	
Example for Publisher / Sender	GMSEC	MSSN	SAT1	RESP	DIR	TLM3	1	1
Example for Publisher / Sender	GMSEC	MSSN	SAT1	RESP	DIR	TLM2	4	4
Example for Subscriber / Receiver	GMSEC	MSSN	SAT1	RESP	DIR	TLM2	*	*

Table 3-19. Properties of the *Miscellaneous Elements* for the Directive Response Message

<i>Miscellaneous Element</i>	Required / Optional	Description	Field Origination in Msg, if applicable
ME1	Required	Component name of Requestor	Echo of "COMPONENT" in header of Request msg
ME2	Required	Status type supplied by Responder	"RESPONSE-STATUS" from content of Response message

Examples:

Two components, APP1 and TLM2, interact with the Directive Response message.

TLM2 subject to send the Directive Response to APP1:

GMSEC.MSSN.SAT1.RESP.DIR.APP1.1

APP1 subscribes to receive its own Directive Response Messages:

GMSEC.MSSN.SAT1.RESP.DIR.APP1.* or

GMSEC.*.*.RESP.DIR.APP1.>

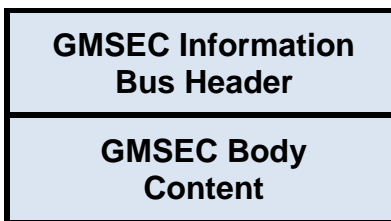
Section 4 GMSEC Standard Message Definition

This section provides an overview of the GMSEC defined messages. This includes a discussion on the general message format, fields, and information about those fields. Within this discussion the GMSEC Information Bus Header is defined and explained. Next, the interactions between the different message types are described. That is, the protocol of interaction is discussed between the Request-Response message pairs, the Request-Response-Message triad, and the Product Request - Product Response - Product Message triad. Finally, a discussion on the categorization of the messages, components, and products is provided that includes frameworks in the form of a tables for categorizing these entities.

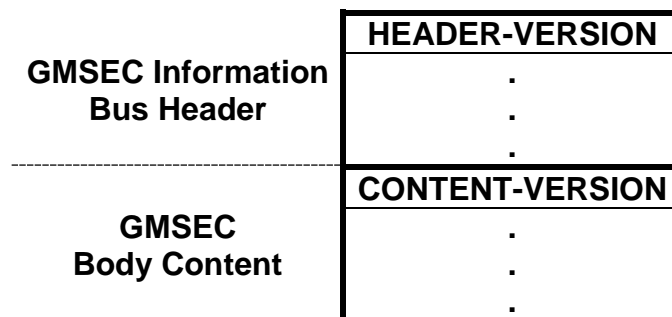
4.1 Format of Message Definitions

4.1.1 General Message Format

Each GMSEC defined message consists of two portions: a GMSEC Information Bus Header and a message Body Content (or, simply Content) portion.



Each of these two message portions contains a version number. The version number identifies the iteration of the message definition and is shown below in the expanded format.



Both the GMSEC Information Bus Header and the message Body Content portion of the message definitions are presented in the format of the table below.

Table 4-1. General Message Description Format

Field Name	Req/ Opt/ A	Value	Type	Notes
VERSION	R			Version Number for this message description
Field name a	R			
Field name b	O			
Field name c	A			

The message definition format contains Field Names, Required/Optional/API field indicator, Value, Type, and Notes. The next several sections describe how these conventions are used to describe the messages.

4.1.1.1 Field Name

The Field Name is the name of an item in the GMSEC Standard Message. It uses visible alphanumeric characters as readable descriptors. The Applications Programming Interface (API) uses the Field Name to pack and unpack messages.

A generalized field-naming convention has been implemented that adheres to the following simple rules.

- Field names shall consist of alphanumeric characters, dashes ("-"), and dots/periods (".")
- Alpha characters shall be capitalized
- For a repeatable series of field names, the convention shall be as follows:

FIELD-NAME.n.CONTENT

Where, when expanded, would develop as

FIELD-NAME.1.CONTENT
FIELD-NAME.2.CONTENT
FIELD-NAME.3.CONTENT

And so on. "n" starts with "1".

- For a field name series, it shall be preceded by a field name that specifies the number of field names in the series as shown in the following example.

NUM-OF-MNEMONICS (=2 for this example)
MNEMONIC.n.NAME
MNEMONIC.n.STATUS
MNEMONIC.n.NUM-OF-SAMPLES (=3 for each mnemonic in this example)
MNEMONIC.n.SAMPLE.n.TIME-STAMP
MNEMONIC.n.SAMPLE.n.RAW-VALUE

Where “NUM-OF-“ is the standard prefix of the field name. The two suffixes in this example, “MNEMONICS” and “SAMPLES”, are then used in the singular form to describe the series of field names that follows. In the above example, the field names would expand as follows:

NUM-OF-MNEMONICS
MNEMONIC.1.NAME
MNEMONIC.1.STATUS
MNEMONIC.1.NUM-OF-SAMPLES
MNEMONIC.1.SAMPLE.1.TIME-STAMP
MNEMONIC.1.SAMPLE.1.RAW-VALUE
MNEMONIC.1.SAMPLE.2.TIME-STAMP
MNEMONIC.1.SAMPLE.2.RAW-VALUE
MNEMONIC.1.SAMPLE.3.TIME-STAMP
MNEMONIC.1.SAMPLE.3.RAW-VALUE
MNEMONIC.2.NAME
MNEMONIC.2.STATUS
MNEMONIC.2.NUM-OF-SAMPLES
MNEMONIC.2.SAMPLE.1.TIME-STAMP
MNEMONIC.2.SAMPLE.1.RAW-VALUE
MNEMONIC.2.SAMPLE.2.TIME-STAMP
MNEMONIC.2.SAMPLE.2.RAW-VALUE
MNEMONIC.2.SAMPLE.3.TIME-STAMP
MNEMONIC.2.SAMPLE.3.RAW-VALUE

Note that “n” starts with “1”.

- For each message, all field names shall be unique.

4.1.1.2 Required, Optional, and API Field Indicator

Some Field Names are Required (R) for processing and others are identified as being Optional (O), Dependent (D), or API (A). The required fields must be present in order to be compliant with the GMSEC Standard Message Definition. An optional Field Name is a placeholder for additional data. The Optional fields may be useful to the Receiver and may be implemented as necessary. Software components, missions, or interface definitions may determine if these fields are required for their particular needs and applications. If a Field Name is listed as Dependent, that Field Name is actually required for a certain Message Type, or is dependent on another field being present. This information will be documented in the specific Message Type section where applicable. The API fields are those filled in by the API software and will overwrite any user supplied data in these fields.

4.1.1.3 Value

Some Fields must contain specific values in order to be GMSEC compliant. For these cases, the valid values are listed in the message content's Value column. If no value is specified, the value of the Field Name is variable; however, it must conform to the specified Type. See the Type description in the next section.

4.1.1.4 Type

The Field Type is the data type. Cross-platform compatibility is achieved using the defined field types listed below. The intention is for the API to perform any necessary type handling such that the types used in the add/get field functions are whatever is native to the language/architecture being used. Therefore the client application will not have to deal with byte-swapping or other number format changes. Type definitions are based on the Institute of Electrical and Electronics Engineers (IEEE) standards. Time field types are based on the ISO 8601 standards.

THIS PAGE INTENTIONALLY LEFT BLANK.

Table 4-2. Field Type Definitions

Field Type	Definition	Range/Comments
Binary [Blob]	0 or more of any combination of bytes, integers, floating points, doubles, time, and strings.	Its structure may be dependent upon message type, message subtype, or application generating the message.
Boolean (1)	False/true, no/yes	[0, 1]
Character	Native single ASCII character representation	[-128, 127]
F32 Float (3)	32-bit single precision floating point representation	32 bits composed of 23 bits for the fraction, 8 bits for the exponent, and 1 sign bit. (See IEEE 754)
F64 Double (3)	64-bit double precision (extended) floating point representation	64 bits composed of 52 bits for the fraction, 11 bits for the exponent, and 1 sign bit. (See IEEE 754)
Header string	Any combination of an UPPERCASE alphanumeric, "-" (dash), and "_" (underscore) characters.	This field type requires fields also used as message subject elements to be uniformly UPPERCASE
I16 Short (3)	16-bit signed integer representation	$[-2^{15}, 2^{15} - 1]$ 16 bits composed of 15 bits for the number and 1 bit for the sign.
I32 Long (3)	32-bit signed integer representation	$[-2^{31}, 2^{31} - 1]$ 32 bits composed of 31 bits for the number and 1 bit for the sign
I64 Longlong (2,3)	64-bit signed integer representation	$[-2^{63}, 2^{63} - 1]$ 64 bits composed of 63 bits for the number and 1 bit for the sign
String	0 or more ASCII characters	Also, see Header string.
Time	String representation of time	See Table 4-1. Ordinal Date and Time Field Type Definition
U16 UShort (3)	16-bit unsigned integer representation	$[0, 2^{16} - 1]$ 16 bits, no sign bit
U32 ULong (3)	32-bit unsigned integer representation	$[0, 2^{32} - 1]$ 32 bits, no sign bit
U64 ULonglong (3)	64-bit unsigned integer representation	$[0, 2^{64} - 1]$ 64 bits, no sign bit
Variable	Field could be any data type	User needs to ascertain the data type of the field prior to accessing the value (e.g. with a function call)

Notes:

1. "Boolean"

The GMSEC Interface Specification always defines the value of the Boolean field to be 0 (zero) or 1. The description or meaning of the value can take various forms, such as no/yes, false/true, disabled/enabled, in-limits/out-of-limits, active/static, and so on. It is important to take into account that some programming and scripting languages (e.g., Java), schemas, commercial products, and custom software will only interpret the value of a Boolean field to be false/true.

2. Field types larger than 32 bits may not be available on 32-bit architecture platforms.

3. In support of both 32-bit and 64-bit architecture platforms for various equipment manufacturers, these ambiguous terms are targeted for future deprecation.

The following tables describe some of the commonly used time formats. The “Time” format is the only data type specified in GMSEC messages. Other formats are included for reference. Time formats generally are UTC based.

Table 4-3. Ordinal Date and Time Field Type Definition

Field Type	Definition	Range
Time	<p>Note: This time type can represent either an absolute or relative time.</p> <p>Time in the form:</p> <p>[+, -] YYYY-DDD-hh:mm:ss[.ff...]</p> <p>Where:</p> <p>“+” and “-” are leading signs used for relative (duration) times. If the sign is omitted, absolute time is indicated. Relative (duration) times are given in the same format, but leading fields may be omitted rather than being set to zeros. For example +0000-000-00:12:21 may be abbreviated +12:21</p> <p>Absolute times are UTC based.</p>	
	"YYYY" is the year.	0000 to 9999
	"DDD" is the Julian day, or day of year.	001 to 365/366
	"hh" is hours of day on a 24-hour clock.	00 to 23
	"mm" is the minutes of hour.	00 to 59
	"ss" is the seconds of minute.	00 to 60 (allowing for leap seconds)
	"ff..." is the base-ten fractional seconds. Fractional seconds are considered optional. If present, the number of digits can vary in length from 1 to 6.	0 to 999999
	Applications need to convert from the string format to native system representations.	

4.1.2 GMSEC Information Bus Header

Table 4-4. GMSEC Information Bus Header

Field Name	Req/ Opt/ API	Value		Type	Notes
Message Identification Information					
HEADER-VERSION	R	2010		F32	Version Number for this message description.
MESSAGE-TYPE	R	Value	Description	Header string	Message type identifier: REQ, RESP, or MSG
		REQ	Request		
		RESP	Response		
		MSG	Message		
MESSAGE-SUBTYPE	R	See Table 3-8. Descriptions of the Message Elements of the GMSEC Message Subject		Header string	Unique message subtype identifier, fixed for GMSEC Standard Messages
MESSAGE-CLASS	O			Header string	Generic field for missions to classify their message set to aid message disposition.
UNIQUE-ID	A			Header string	Globally unique ID supplied by the API
PUBLISH-TIME	A			Time	Time the message was published. Supplied by the API
Connection Information					
MW-INFO	A			String	Container for information on the underlying middleware. Content is middleware specific.
CONNECTION-ID	A			U32	Unique ID for each connection per process. Supplied by the API
Mission Information					
MISSION-ID	R			Header string	Unique mission name, e.g., MOONMAP, SUNSCAN, TEMPTRACK, etc
CONSTELLATION-ID	O			Header string	Used for constellations or satellite groupings
SAT-ID-PHYSICAL	O			Header string	An ID for the satellite that is fixed for its mission life
SAT-ID-LOGICAL	O			Header string	An ID for a single or group of satellites that can change during its mission life (ex., a positional reference)
Component Information – Location and Identification					
FACILITY	R			Header string	A physical source (i.e. ACS Lab, CandDH String, etc.) generating the message, could be spacecraft, remote site, etc.
NODE	A			Header string	Actual device (host) generating the message. Supplied by the API.
PROCESS-ID	A			I16	Application ID for Onboard Events or Process ID for Ground Events. Supplied by the API
Component Information – Logical					
CLASS	O			Header string	See Table 4-28. Software Class and Subclass Categories
COMPONENT	R			Header string	Name of software application, ex. APP1, CLIENT2, TAC3
SUBCOMPONENT1	O			Header string	First subsystem level within the component that produced the message
SUBCOMPONENT2	O			Header string	Second subsystem level within the component that produced the message
USER-NAME	A			Header string	Account name or owner of the account that started the component
ROLE	O	Value	Description	I16	Role the component is assigned in the configuration (Primary, Backup, Hot
		1	Primary, Master		

		2	Secondary, Backup		Backup, Secondary, Spare ...). Roles are dependent on the operational concepts being employed in the configuration.
		3	Tertiary		
		...	Spare, ...		

4.1.2.1 Notes on Fields in the Information Bus Header

4.1.2.1.1 Fields Used for Message Subjects

MISSION-ID, CONSTELLATION-ID, SAT-ID-PHYSICAL, SAT-ID-LOGICAL, MESSAGE-TYPE, and MESSAGE-SUBTYPE:

From the discussion in Section 3.3 Standards of GMSEC Message Subject Names it was noted that the first few elements of the GMSEC subject are fixed. Please refer to this section for important information on the format of subject names and their content. The first few elements of the GMSEC message subject can be made up from fields in the GMSEC Information Bus Header. The format of the fixed portion of the GMSEC Message Subject is given in the following table:

Table 4-5. Fixed Portion of the GMSEC Message Subject

	Subject Standard	Mission Elements		Message Elements	
Subject Element	Specification	Mission	Sat ID	Type	Subtype
	FIXED PORTION				

Recall that the definition of the subject follows the following format:

SPECIFICATION.MISSION.SATID.MSGTYPE.MSGSUBTYPE.*ME1.ME2.ME3...*

The following table shows how fields from the GMSEC Information Bus Header can be directly used as elements of the GMSEC Message Subject. It is important to remember that the GMSEC Message Subject is text that is in UPPERCASE, so that if a field is extracted from the Information Bus Header and used in the Message Subject, it must be in UPPERCASE.

Table 4-6. Mapping of Information Bus Header Fields to the GMSEC Message Subject

Message Subject Element	Possible Fields Used from the Information Bus Header Field
Specification	None
Mission	MISSION-ID or CONSTELLATION-ID
SAT ID	SAT-ID-PHYSICAL or SAT-ID-LOGICAL
Type	MESSAGE-TYPE
Subtype	MESSAGE-SUBTYPE

4.1.2.1.2 Fields for Message Uniqueness

UNIQUE-ID, PUBLISH-TIME:

The UNIQUE-ID is a globally unique ID supplied by the API. It will make each message uniquely singular from all others. The PUBLISH-TIME is likewise supplied by the API at the time of publication of the message. These fields are valuable for identification and tracking purposes.

4.1.2.1.3 Middleware Tracking Information

MW-INFO, CONNECTION-ID:

The GMSEC API will also fill in these Information Bus Header fields prior to sending the message. They serve to identify the connection and/or path of the message with the underlying middleware.

4.1.2.1.4 Component Information – Location and ID

FACILITY, NODE, PROCESS-ID:

These refer to physical locations, equipment, or identifiers. For example facility might refer to a city, building, LAN, satellite control center, room, or whatever is used to uniquely identify the physical location. NODE will normally be used to identify a single piece of equipment, commonly a computer. It could also refer to a larger entity such as a satellite tracking station. Typically, the node is found within the facility. Another example, are the computers (nodes) found on a LAN (facility).

PROCESS-ID is the unique ID of an executing task or process on a NODE usually assigned by the host operating system. The combination of these 3 fields will serve to specifically identify the executing software process within an enterprise. The API will supply the NODE and PROCESS-ID information on which the process is executing in the Information Bus Header.

4.1.2.1.5 Component Information – Logical

CLASS, COMPONENT, SUBCOMPONENT1, SUBCOMPONENT2, USER-NAME, ROLE:

These fields provide for source traceability of a message. A GMSEC Class is a high level category of functionality. A GMSEC Subclass is defined as a subset genre of a Class. A Class is composed of one or more subclasses. A GMSEC Component is defined as the name of the executing software application that fulfills in part or in whole, an instance of a GMSEC Class or Subclass. Class and Subclass are sometimes used interchangeably to refer to a type of system, whereas component always refers to an actual piece of software. The component generates GMSEC messages and identifies in the Information Bus Header the Class and/or Subclass to which it belongs. It has the option to further delineate the source of a message by using the SUBCOMPONENT1 and SUBCOMPONENT2 fields. The relationship between the GMSEC Classes and Subclasses is shown in Table 4-28. Software Class and Subclass Categories.

The GMSEC API will supply the USER-NAME field which is the name or owner of the account that started the component. The ROLE of the component is optionally supplied to indicate what function the component is assigned to play in the overall concept of operations. Components will perform different functions depending on their roles and responsibilities.

4.1.3 Message Contents

The Message Contents for each message type will vary. A CONTENT-VERSION is required for each message in order to map compliance to a version of this document. Each message type will have a combination of required and optional fields, each of a variety of data types.

Table 4-7. Format of the Message Content Portion

Field Name	Req/ Opt	Value	Type	Notes
CONTENT-VERSION	R		F32	Version Number for this message content description
UNIQUE-ID	A		Header String	Unique ID used distinguish the message
Field name 0-m	R		Specific to message type	
Field name 0-n	O		Specific to message type	

4.1.3.2 Other Fields in the Message Content Portion

A mission or component is not restricted to using the GMSEC defined fields in the content portion of the message. A mission or component can enhance the GMSEC message definition with additional fields of their own definition to create their own enhanced mission specific message definitions.

4.2 GMSEC Messages: Their Characteristics and Interactions

4.2.1 GMSEC Message Type Overview

Three types or classes of messages have been defined within the GMSEC Architecture. The type of the message identifies the kind of communication for which the actual message is being used. The three types of messages are:

- **Message** (MSG) or generic message
- **Request** (REQ) message
- **Response** (RESP) message

Typically, the generic **Message** is published without a required or expected response (though it may cause an action when received). The **Request** message is used to request a specific action or information from a service or data provider, or product generator. The **Request** message may or may not require a reply message. If so, the **Response** message is used.

In the GMSEC Message Subject Name Definition, the type of message (**MSG**, **REQ**, and **RESP**) is specified in the **Type** element.

Table 4-8. GMSEC Message Subject Name Definition

Subject Element	Subject Standard	Mission Elements		Message Elements		Miscellaneous Elements			
	Specification	Mission	Sat ID	Type	Subtype	ME1	ME2	ME3	ME4 ...
	FIXED PORTION					VARIABLE PORTION			
	Required Elements					Message definition determines whether a Miscellaneous Element is required or optional			

These three fundamental GMSEC message types can be used in various combinations with one another to create an infinite number of message exchange patterns. In turn, these message exchange patterns can be used in the description of the interfaces for any number of services. A basic set of GMSEC message exchange patterns (MEPs) are described later in this section.

In the GMSEC message type discussion that immediately follows, the terms **REQ**, **RESP**, and **MSG** will be used in reference to the type of message. The terms message, request, and response will refer to actual messages.

4.2.1.1 GMSEC MSG Message Type

The MSG message type is the basic message type used to convey any kind of information. This information can be normative or critical. It can be used by itself or in combination with the other message types of REQ and RESP. When used by itself as a single message it is commonly sent for informational purposes. Perhaps the two most common uses of the MSG type are the GMSEC Log message and the Component-to-Component Transfer (C2CX) Heartbeat message. The Log message is primarily informational. It is typically published / sent by all components with no further regard or accounting by the sender. Other components will subscribe to the Log messages; some without regard to their source. No requests are made for the messages and no direct linkage exists between the publishers and subscribers.

Likewise, the C2CX Heartbeat message is pumped out periodically with no regard for its destination or subscriber. Other components will subscribe to the Heartbeat messages and use them to monitor the active status of the publishers. Again, there is no request made for the messages, no direct linkage exists between the senders and subscribers, and no interest on the part of the sender to receive a response or feedback.

A final example of the uncoupled use of the MSG message type is as a stream of data messages, most typically for a telemetry data stream. In this circumstance, one message follows another in a continuous stream of MSG messages. The stream of MSG messages can be solicited or unsolicited. An example of unsolicited MSG messages is a real-time telemetry data provider that automatically publishes CCSDS formatted frames or packets onto the network upon receiving them from a front end provider. A solicited MSG data stream can occur following a Request/Response message exchange for a replay of stored telemetry data.

When the MSG type is used in conjunction with the other message types of REQ and RESP, it will either precede (instigate) a Request/Response message exchange, or follow the exchange such as:

- MSG-REQ-RESP
- REQ-RESP-MSG

The details of these Message Exchange Patterns are discussed in later sections.

4.2.1.2 GMSEC REQ Message Type

The REQ message type is typically used to make a request of another component. The request could be for data, a product, service, or to take some action. The request may or may not require a response. For example, one component may request another component to take some action and does not

need to immediately know if the action was successful or not, as it may not care or it may discover the result by other means, or it may take a considerable amount of time for the action to be completed.

A common use of the REQ message type will be in conjunction with the Directive Request message. The Directive Request message typically asks another component (in its own language syntax, or perhaps an operations language meta-model) to perform some action. If a response is required, it is specified within the content of the Directive Request message.

4.2.1.3 GMSEC RESP Message Type

The RESP message type is used in response to a REQ message type. It is never used in the initiation of a message exchange, only in response. For example, when sending a Directive Response message, the TYPE and SUBTYPE elements of the message subject will be RESP and DIR, respectively.

Within all response messages is a “Response Status” substructure to provide status information on the request. Depending on the status code (in the RESPONSE-STATUS field) within the structure, the response message can take on a number of different meanings. Further information on the status of the request can be found in the optional RETURN-VALUE field of the structure. This structure and the possible status codes are shown in the following table.

Table 4-9. Response Status Substructure

STRUCTURE: Response Status				
RESPONSE-STATUS	R	Value	Description	I16
		1	Acknowledgement	
		2	Working / Keep Alive	
		3	Successful Completion	
		4	Failed Completion	
		5	Invalid Request	
		6	Final Message	
TIME-COMPLETED	O			Time
RETURN-VALUE	O			I32

Depending on the type of messages and the service/data the responder is providing, there may be a need for more than one Response message to be returned to the requestor. For example, the responder may initially return a Response message with a RESPONSE-STATUS of “Working/Keep Alive”. This status indicates the message has been received, the request is still active or being processed, and is not yet complete. The responder may determine to periodically return this same status any number of times until the processing has been completed. At this point a Response message is returned with a status of either “Successful” or “Failed” in the RESPONSE-STATUS field. Thus, a series

of Response messages may be returned to the requestor. Each return status is explained in greater detail below. Due to the variable nature of time required to process requests, there is no defined interval between the Request and (multiple) Response messages. Discussion of each status code follows.

- **Acknowledgement**
 - **Meaning:** The Request Message was received. No action has yet been taken on the Request Message.
 - **Sequencing:** This could be the first of a series of Response messages, or be the one and only final message in the case where the message exchange pattern is Request/ACK. Only 1 ACK status would normally be returned.
- **Working/Keep Alive**
 - **Meaning:** The request has been received and is actively being processed.
 - **Sequencing:** This status could initially be returned or could follow an ACK status (be the 2nd). This status could be returned a multiple number of times. It should not be the last response message returned.
- **Successful Completion:**
 - **Meaning:** The request was valid and has been processed in a successful manner.
 - **Sequencing:** This status could be initially returned, or it could follow either of the two statuses above. It will also be the last status returned. If initially returned, the responder was able to complete the request in a timely manner and immediately return a response. In other cases, the responder required a more lengthy time period to complete the request.
- **Failed Completion:**
 - **Meaning:** The request was valid, processing was initiated, but the responder was unable for any number of reasons to fully and successfully complete the request.
 - **Sequencing:** This status could initially be returned (the 1st and only status), or could follow an ACK or Working/Keep Alive status. It will not follow a Successful status. It will be the last status returned.
- **Invalid Request:**
 - **Meaning:** The request message was unable to be fully interpreted for processing. There could be missing parameters, inconsistencies, or incorrect values.
 - **Sequencing:** This status could be the first and only one returned. It could also follow an ACK. It will be the last status returned.

Final Message:

- **Meaning:** This is the last and final message in a series of messages. This status code has been included to provide an unmistakable indication that this is indeed the final message in a series.
- **Sequencing:** This status will never be the first one returned and will always follow previous Response messages, either a series of data value messages, or other Response messages.

A summary of the sequencing of the statuses is shown in the following table.

Table 4-10. Sequence of Response Status

Status Code		Sequence		
		Initial Status	Intermediate Status	Final Status
1	Acknowledge	Yes	No	Yes
2	Working/Keep Alive	Yes	Yes	No
3	Successful Completion	Yes	No	Yes
4	Failed Completion	Yes	No	Yes
5	Invalid Request	Yes	No	Yes
6	Final Message	No	No	Yes

4.2.1.5 GMSEC API Transport Mechanisms

4.2.1.5.1 GMSEC API Send and Receive Functions

For the following discussion please refer to the GMSEC API User's Guide documentation for additional detail on the behavior of the API functions. Also, the following discussion uses these similar, yet distinct terms:

- REQ – one of the three types of GMSEC messages
- Request message – a GMSEC standard (defined) message
- Request function – an API function used to send a message

The following table shows that there are three callable functions available within the GMSEC API to send messages. They are the “**Request**”, “**Reply**”, and “**Publish**” functions. While the “**Publish**” function can be used to send any type of message, the “**Request**” function should only be used to send REQ type messages and the “**Reply**” function should only be used to send RESP type messages.

Table 4-11. API Send Functions Used with the GMSEC Message Types

Message Type	API Send Functions		
	Request	Reply	Publish
REQ	√		√
RESP		√	√
MSG			√

The next table shows that there are primarily two mechanisms within the GMSEC API available to receive messages. They are the paired functions of “**subscribe/receive**” and the special case of the “**request**” function. The “**subscribe/receive**” function pair can be used to receive and process any type of message. The “**request**” function is unique in that it can be used to both send and receive a message. When components interact using the API Request/Reply functions, the requestor can first send a REQ message type with the API Request function, and then wait (block) for the RESP message type. In this way the API Request function essentially has a dual purpose to both send a Request message and receive a Response message.

Table 4-12. API Receive Functions Used with the GMSEC Message Types

Message Type	API Receive Functions	
	request	subscribe / subscribe
REQ		√
RESP	√ *	√
MSG		√

* User must initially send a Request Message and wait (block) for the Response Message.

Other mechanisms are also available within the GMSEC API to retrieve messages automatically. More specifically, a user can utilize GMSEC API functions to have messages automatically dispatched from an incoming queue rather than pick them off one-by-one with the “receive” function. Please refer to the GMSEC API User’s Guide for further details.

4.2.1.5.2 Behavior of the GMSEC API Send and Receive Functions

In many situations, two applications will interact using the GMSEC Request and Response message pair. These message pairs include the Directive, Mnemonic Value, Replay Telemetry, and others. This section describes the different behaviors of the GMSEC API send and receive functions when using the Request and Response messages. Detailed information on the use of the API can also be obtained from the GMSEC API User’s Guide documentation.

The Request/Response message interaction typically involves a point-to-point interaction between two components. One component acts as the data/product/service requestor (or service consumer). The other component acts as the data/product/service responder (or service provider). The data requestor will use the Request Message and the data responder will use the Response Message.

As described in the previous section, two delivery mechanisms are available within the GMSEC API for the Request/Response exchange of messages. They are the:

- 1) API **request** and **reply** functions, and
- 2) API **publish** and **subscribe/receive**.

4.2.1.5.2.1 GMSEC API Request and Reply Functions

The API **request** and **reply** functions have been built for point-to-point message exchanges. The use of **request/reply** is intended for the private exchange of messages with known subjects between two or more applications. The requesting component will use the API **request** function to send the specific GMSEC Request Message to the responder. The responding component will use the API **reply** function to send the corresponding GMSEC Response Message back to the requestor. Multiple Response Messages could be returned with this mechanism. The API **reply** function requires, as parameters, a received message and a message for the reply. See the following sequence diagram that illustrates this message exchange pattern using the API Request and Reply functions.

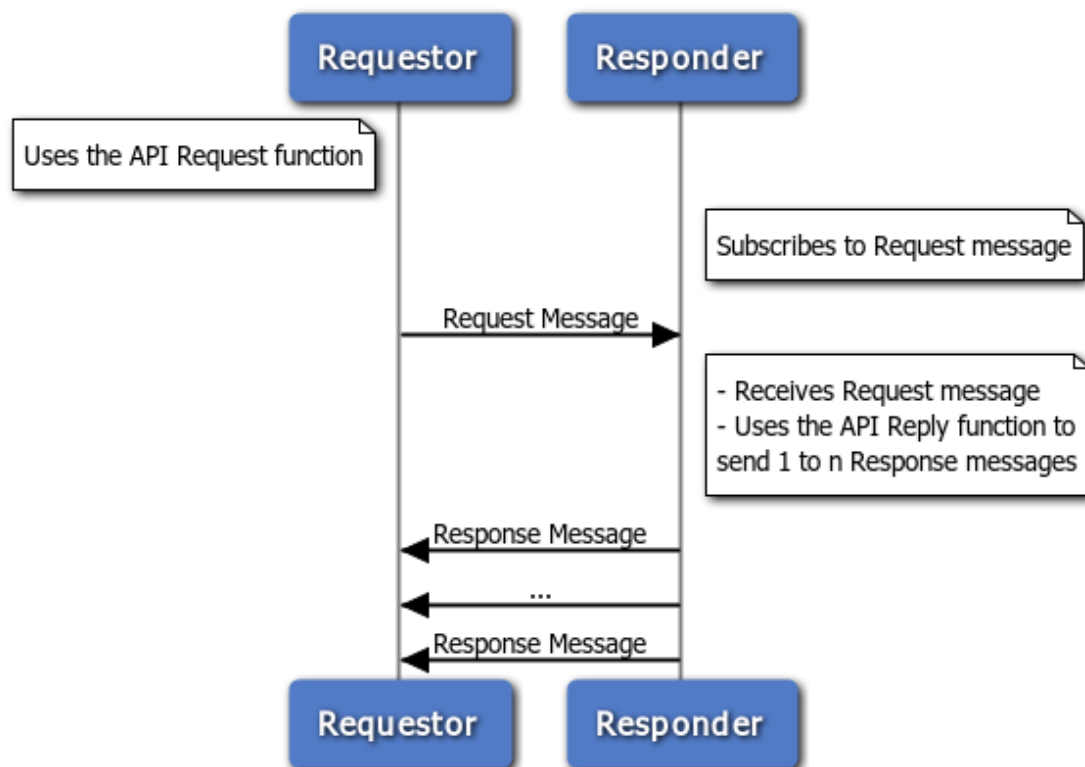


Figure 4.2.1.5.2.1-1 Sequence Diagram of Request-Response Message Exchange Using the API Request and Reply Functions

4.2.1.5.2.2 GMSEC API publish and subscribe/receive Functions

An alternative methodology to sending and receiving the Request and Response messages is with the API **publish** and **subscribe/receive** functions. In this message exchange the API **publish** function is used by the requestor to send the Request message and also used by the responder to send the Response message. The **subscribe** and **receive** functions are used to receive the messages. A message must first be subscribed to before it can be received.

4.3 GMSEC Message Exchange Patterns

The previous sections described some specific ways in which the three GMSEC message types of REQ, RESP, and MSG can be used. In fact, these three message types could be endlessly combined to create an endless number of message exchange patterns (MEPs). Fortunately, a limited number of MEPs can be identified to satisfy practically all interaction requirements of software applications including service consumers and providers. Four (4) basic MEPs are identified and categorized in the following table.

Table 4-13. GMSEC Message Exchange Pattern Types

Exchange Pattern	Description	Messages Types and Sequence
1. Publish	Send a single message	<ul style="list-style-type: none"> A single MSG or REQ message
2. Request/Response	Send a request and receive response message(s)	<p><u>Single Response</u></p> <ul style="list-style-type: none"> REQ/ACK REQ/RESP <p><u>Multiple Responses</u></p> <ul style="list-style-type: none"> REQ/ACK/RESP REQ/ACK/Interim Status.../RESP REQ/Interim Status.../RESP <p>(Interim Status is a Response message with a value of "Working" in the RESPONSE-STATUS field)</p>
3. Triad of Request, Response, and Publish	Use all three types of messages	<ul style="list-style-type: none"> Triad 1: <ul style="list-style-type: none"> REQ/RESP/MSG Triad 2: <ul style="list-style-type: none"> MSG/REQ/RESP
4. Subscription <ul style="list-style-type: none"> Subscribe Data Series Unsubscribe 	Subscribe to data, receive the data, and unsubscribe	<ul style="list-style-type: none"> REQ/RESP (Subscribe), MSG ... (Data Series), REQ/RESP (Unsubscribe)

Each of the 4 MEPs is summarized in the following table. Following the table, each MEP is further detailed in its own section that includes a description, usage, and a sequence diagram depicting the interactions.

Table 4-14. GMSEC Message Exchange Patterns

Pattern	#	Description / Use	MSG Sequence Direction (Wrt Initiator) and Message Types Used	Fault Message	Examples
Publish	1.	Publish a single message, for any purpose, with no follow up required	Out: MSG or REQ	None	1. Send a Log or Heartbeat message 2. Send a Directive to a component for execution; no response is necessary
Request / Response Theme (2.1 – 2.5)					
Request / ACK	2.1	Publish a message and receive an acknowledgement	Out: REQ In: RESP (ACK)	None	1. A component sends a request and needs to know if it was received. 2. One component pings other components to test their responsiveness
Request/Response	2.2	For Requests that can be fulfilled with a single Response message	Out: REQ In: RESP (Status)	Response (Status)	Request a product, data, or a service from another component and receive the result in the single RESP message.
Request/ACK/Response	2.3	The requestor requires an acknowledgement to the Request, then a Response.	Out: REQ In: RESP (ACK) In: RESP (Status)	Response (Status)	The initial Request message is responded to with a Response message having a status = ACK; then the Response message (with appropriate status) will follow.
Request/ACK / Interim Status/ Response	2.4	For requests that take an extended time, initially provide an ACK to the Request, then periodic status updates, and then the final response message.	Out: REQ In: RESP (ACK) In... RESP (Working) ... In: RESP (Status)	Response (Status)	A request for a product is responded to with an ACK, then any number of “working” messages as the product is generated, ending with a final Response message containing the product.

Pattern	#	Description / Use	MSG Sequence Direction (Wrt Initiator) and Message Types Used	Fault Message	Examples
Request/ Interim Status/ Response	2.5	Identical to the Request/ ACK /Interim Status/Response pattern but with no ACK message.	Out: REQ In-in... RESP (Working) ... In: RESP (Status)	Response (Status)	A request for a product is responded to with any number of "working" messages as the product is generated, ending with a final RESP message containing the product.
Triad Theme (3.1 – 3.2)					
TRIAD 1 Request/ Response/ Publish	3.1	For requests that either take a long time, or that require a subsequent message, and no interim status updates are required. (Combination of Request/Response and Publish). Send notification that Requests can be accepted. Then accept request(s) for that product/service. (Combination of Publish and Request/Response).	Out: REQ In: RESP(Working or Status) In: RESP or MSG	Response (Status)	A request for a product is responded to with the RESP message. Later, when the product is generated or made available, it is sent with a RESP or MSG type. The requestor does not require any interim status messages.
TRIAD 2 Publish/ Request/ Response	3.2		Out: MSG In: REQ Out: RESP (Status)	Response (Status)	Provider sends message announcing availability of product. Consumers use the Request/Response interaction pattern to then request and receive the product.
Subscription (4)					
Subscription • Subscribe • Data stream • Unsubscribe	4.	Subscribe and unsubscribe to data, products, or message streams. (Combination of Request/Response, Publish and Request/Response.)	Out-In: REQ - RESP (Status) In... MSG ... Out-in: REQ - RESP (Status)	Response (Status)	Use Request/Response to subscribe to a series of messages such as a telemetry data stream or mnemonic set. The stream is subscribed to and ingested. Later, unsubscribe to the messages.

See the **Table Legend** that follows:

Table Legend

Term	Meaning
Wrt	with respect to
REQ, RESP, and MSG	Are message types
"..."	indicates any number of these messages from 0 to n
ACK, Working, Status	<ul style="list-style-type: none"> RESP (ACK): indicates an ACK message in the form of a Response message with a status code of "ACK" in the RESPONSE-STATUS field. RESP (Working): indicates an interim status message in the form of a Response message with a status code of "Working" in the RESPONSE-STATUS field. RESP (Status): indicates a final Response message in the message exchange pattern. <ul style="list-style-type: none"> For successful exchanges, the status code is either "Success" or "Final Message" in the RESPONSE-STATUS field. For fault messages, the status code is either "Failure" or "Invalid" in the RESPONSE-STATUS field.

4.3.1 MEP Publish

4.3.1.1 Description

The simplest message exchange pattern involves no obvious exchange with the sender. The sender publishes a message, and from the perspective of the sender, the exchange is complete. The publisher has no more interest in the message and has no need to follow up on its progress. Some other component will subscribe to the message and process it according to the subscriber's requirements.

Either of two GMSEC message types can be used for this pattern; either a GMSEC MSG type or REQ type message. In the case of a REQ message type, the sender may request an action but not require knowledge of the result via a Response message which will be indicated in the Request message.

4.3.1.2 Usage

Most typically, this pattern is seen when a software component publishes a GMSEC Log message using the MSG message type. The Log message and this exchange pattern provide a simple, one-way means of information distribution.

Also, data streams of telemetry will be published using the GMSEC MSG type. Again, the data is published unidirectional with subscribers ingesting the messages and processing them as programmed.

A second message type, REQ, can also be used for this one-way pattern. In this case, a component will send a Request message but has no need for follow up. No acknowledgement or response is required. An indicator is present in the Directive Request message on whether or not a response is required.

4.3.1.3 Sequence Diagram



Figure 4.3.1.3-1 Sequence Diagram of Publish Message Exchange Pattern

4.3.2 MEP Request/Response (“Variations on a Theme”)

This section begins the series of variations on the theme of a Request/Response message exchange pattern. Altogether, five (5) Request/Response MEPs are described in the following sections. These MEPs begin with two simple patterns where a single Response message is returned for the Request message. The remaining three (3) MEPs involve multiple Response messages being returned for a single Request message.

4.3.2.1 MEP Request/ACK

4.3.2.1.1 Description

The Request/ACK message exchange pattern consists of a published Request message and a returned Response message. The RESPONSE-STATUS field of the Response message contains a value of “Acknowledgement”. No further messages will be returned to the sender.

4.3.2.1.2 Usage

A component sends a Request message and needs to know if it was received. No further information is necessary. Most likely, the sender of the request will need to take subsequent action if the message was not received. Therefore, the Request/ACK pattern can provide the confirmation the message was received when no other information is necessary.

Another possible usage is to “ping” other components. One component may need to take a roll call of members in its group. This could be accomplished with a Request message (e.g., Directive Request) that requires all recipients to return a Response message with the status of ACK.

4.3.2.1.3 Sequence Diagram

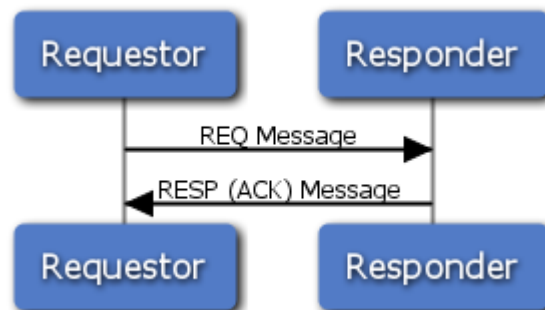


Figure 4.3.2.1.3-1 Sequence Diagram of Request/ACK Message Exchange Pattern

4.3.2.2 MEP Request/Response

4.3.2.2.1 Description

The Request/Response message exchange pattern is a common one-for-one message exchange. This MEP takes the previously described Request/ACK one step further. In this case the Response message will contain an informative status code on the results of the request. See Section 4.2.1.3 GMSEC RESP Message Type for a discussion on the possible status codes for a Response message. The Response message may also contain the resultant data and information, either within the message or referenced. Many GMSEC message definitions are paired in the Request/Response fashion.

4.3.2.2.2 Usage

Components that need to know the results and/or require information from a request will use the Request/Response MEP. Requests can be made for almost anything including the following:

Telemetry and Command

- Replay Telemetry Request/Response
- Mnemonic Value Request/Response
- Archive Mnemonic Value Request/Response
- Command Request/Response

Product and Services

- Product Request/Response

Function Specific

- Directive Request/Response

4.3.2.2.3 Sequence Diagram

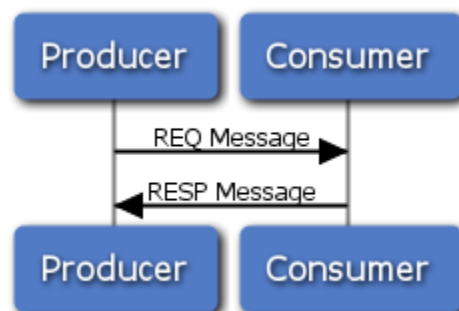


Figure 4.3.2.2.3-1 Sequence Diagram of Request/Response Message Exchange Pattern

4.3.2.3 MEP Request/ACK/Response

4.3.2.3.1 Description

The Request/ACK/Response message exchange pattern is an overlaying combination of the Request/ACK and the Request/Response MEPs. In this case, the requestor requires confirmation that the request was received, final status information on the results of the request, and most likely information generated from processing the request.

4.3.2.3.2 Usage

Usage is similar to the Request/Response MEP.

4.3.2.3.3 Sequence Diagram

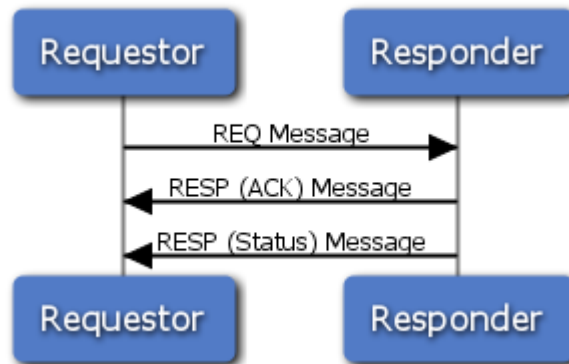


Figure 4.3.2.3.3-1 Sequence Diagram of Request/ACK/Response Message Exchange Pattern

4.3.2.4 MEP Request/ACK/Interim Status/Response

4.3.2.4.1 Description

For some requests, an extended period of time may be required to complete the action or task. Additionally, the requestor may want to be kept abreast of the progress of such a request. In this case, the Request/ACK/Interim Status/Response message exchange pattern is appropriate. This pattern provides for the following responses:

- Initial Response message with an Acknowledgement (ACK) status
- Any number of interim Response messages with a status of "Working/Keep Alive"
- A final Response message with the status of the request and possibly information generated from processing the request.

The number and frequency of the interim Response messages are left up to the interacting components to be determined prior to execution. For example, depending on the request and the resources required by the responder, it could take seconds, minutes, hours, or even days to complete a request. The requestor may want to be kept informed with periodic Response messages (with a status of "Working") to be assured that the request has not been lost, dropped, or forgotten. Thus, the requestor can be kept informed over an extended period of time that a final response is forthcoming. If the interim status Response messages cease, the requestor can determine what subsequent action to take.

4.3.2.4.2 Usage

A component may request a telemetry data product; say an archived data set or a data plot. The provider of this data product may need to first validate the request, and then request the necessary data from another data provider. This could take the form of another Request/Response MEP with another component. Once the data set has been retrieved, the data plot can be generated and finally provided back to the original requestor. In this instance, seconds may transpire while the original request ripples through a system generating other product and service requests.

A second, longer term example is a request made for a product that is not yet available and requires ancillary data that won't be available for some time. For example, a request may be issued for a satellite contact schedule, tracking data, an activity plan, or for the set of available resources. These products may be generated on a scheduled, periodic basis, after the completion of a pass, or only after some other product has been generated in the future. The provider to the original request may store or queue the request to later be acted upon when it is feasible, say when other data products become available. In the meantime, the responder will issue a "Working/Keep Alive" status in interim Response messages, every minute or hour until the request can be satisfied. If some link in

the sequence chain of dependent product generation is broken, i.e. a necessary product is not forthcoming; a fault Response message can be issued. In either case, whether the request can be satisfied or not, a final Response message can be issued and the requestor(s) can determine their respective appropriate actions.

4.3.2.4.3 Sequence Diagram

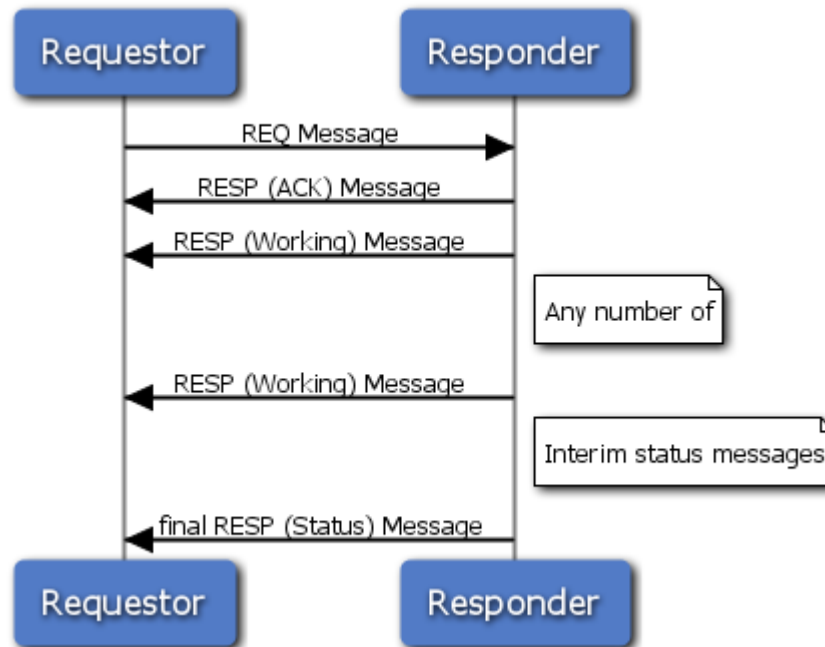


Figure 4.3.2.4.3-1 Sequence Diagram of Request/ACK/Interim Status/Response Message Exchange Pattern

4.3.2.5 MEP Request/Interim Status/Response

4.3.2.5.1 Description

The Request/Interim Status/Response message exchange pattern is an abbreviated form of the Request/ACK/Interim Status/Response MEP. No ACK Response message is provided in the sequence, only interim status messages and a final Response message.

4.3.2.5.2 Usage

Usage is similar to the previous Request/ACK/Interim Status/Response MEP.

4.3.2.5.3 Sequence Diagram

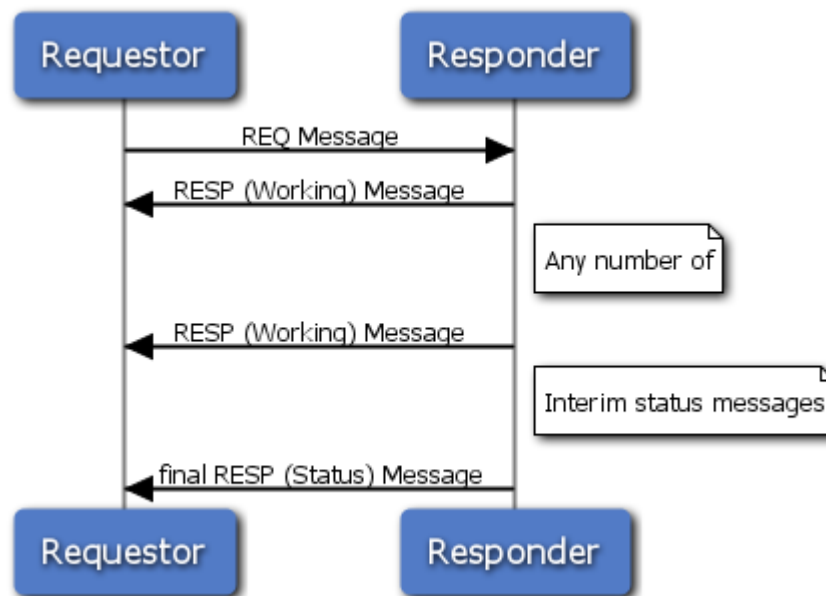


Figure 4.3.2.5.3-1 Sequence Diagram of Request/Interim Status/Response Message Exchange Pattern

4.3.3 MEP Message Triad

In the previous sections the pairing and interaction of the Request and Response messages were discussed. The previous sections provide a foundation for the discussion in this section. This section describes the two (2) Triad message exchange patterns. A message Triad is a message exchange pattern that involves all three types of messages in a specific order. Two message Triad MEPs have been defined. Their order is:

- Triad 1:
 - Request - Response - Publish
- Triad 2:
 - Publish - Request - Response

Triad	Message Exchange Pattern		
	Step 1	Step 2	Step 3
Triad 1 Step: Message Type:	Request REQ	Response RESP	Publish RESP or MSG
Triad 2 Step: Message Type:	Publish MSG	Request REQ	Response RESP

A number of functional message triads have been defined. They include:

Replay Telemetry

- Replay Telemetry Request
- Replay Telemetry Response
- Telemetry Message

Mnemonic Values

- Mnemonic Value Request
- Mnemonic Value Response
- Mnemonic Value Data Message

Archive Mnemonic Values

- Archive Mnemonic Value Request
- Archive Mnemonic Value Response
- Archive Mnemonic Value Data Message

Products

- Product Request
- Product Response
- Product Message

4.3.3.1 MEP Triad 1: Request/Response/Publish

4.3.3.1.1 Description

The Triad 1 message exchange pattern of Request/Response/Publish will typically be used where requests could take a long time to fulfill, or require a subsequent message after the Request/Response interaction. They also do not require any interim status messages. The Triad 1 MEP can be thought of as a combination of the Request/Response and Publish MEPs.

The Triad 1 MEP will be used where the provider of a service or product can respond fairly quickly with an indication that the request can be satisfied, but cannot provide the results within the Response message itself. If the request can be satisfied, by indicating a “Successful” or “Working” status within the Response message, the requestor knows a subsequent message will follow. The subsequent message will contain information about the results of the request. The subsequent message could be a Log message, one of the aforementioned data messages, a Product message, or another Response message that is better suited to contain the requested information. In some cases, only one subsequent message will follow. In other cases, a stream of messages (RESP or MSG) may be required to complete the data request.

Note that this MEP differs from a subscription MEP. A subscription MEP remains open and messages will be published until the subscription is cancelled. The Triad 1 MEP is a one-time request for information, data, service, or product that may take one or more messages to fulfill.

4.3.3.1.2 Usage

A user requests a data product from a product provider. The provider is able to satisfy the request and this is conveyed through the Product Request/Product Response interaction. When the requested product is generated or available, it will be published with the Product Message.

A set of historical mnemonic values is requested from a data provider. The Archive Mnemonic Value Request message allows a number of delivery options, including the option to receive archived mnemonic data as a stream of messages. The subscriber may prefer to receive and process archived data in the same manner as real-time data, i.e. as a stream of messages.

4.3.3.1.3 Sequence Diagram

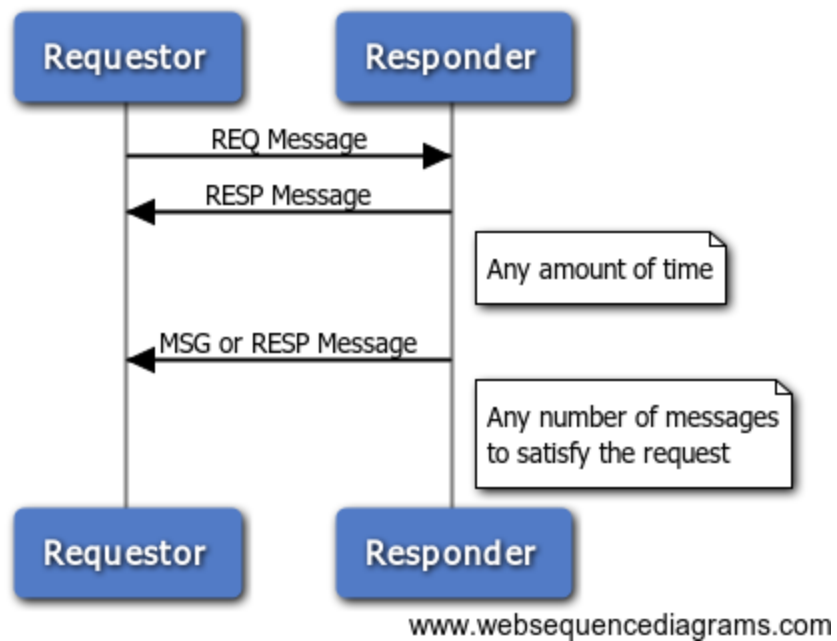


Figure 4.3.3.1.3-1 Sequence Diagram of Triad 1: Request/Response/Publish Message Exchange Pattern

4.3.3.2 MEP Triad 2: Publish/Request/Response

4.3.3.2.1 Description

See the previous section for a discussion on message triads.

This section describes the message exchange pattern Triad of

- Publish – Request - Response

The Triad 2 Publish/Request/Response message exchange pattern is a combination of a Publish MEP and a Request/Response MEP. The initial Publish of a MSG message will instigate a follow up Request/Response interaction. This Publish/Request/Response MEP will typically be initiated by a service or product provider. The data or product provider will publish a MSG message to notify interested subscribers that a product, service or data is now available. Subscribers to the MSG message can then request and receive the product through the Request/Response interaction.

4.3.3.2.2 Usage

A schedule product producer has just completed the compilation of an operational schedule for the next day. It issues a Log message that contains information on the type of product and how/where to acquire it. The schedule execution component has previously subscribed for this particular message. When the message is received and parsed, it issues a Product Request message to the producer for the operational schedule product. The product is received within the Response message and readied for the next day's operations. Note that an alternate methodology for this interaction could be accomplished with a Subscription MEP discussed in the following section. In this case, the producer of the schedule product would provide a subscription service. Parties interested in knowing when a schedule has been generated and is available, could subscribe with the producer. The producer would automatically provide the product when available for any subscriber.

A second scenario could involve a mnemonic data provider. In this example, the data provider has just completed "scrubbing" the data (merging, gap filling, and correcting) and publishes a message to that effect. Users interested in clean data can then initiate the Request/Response interaction for the scrubbed archived mnemonic data.

4.3.3.2.3 Sequence Diagram

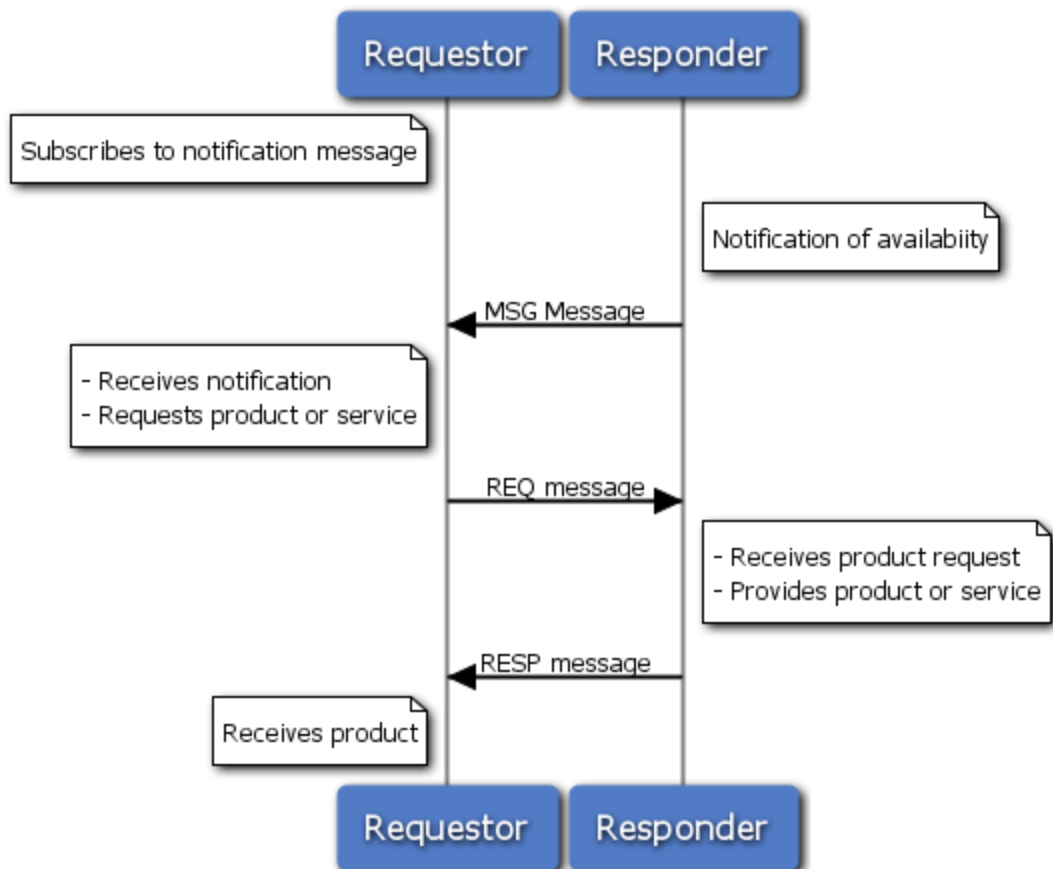


Figure 4.3.3.2.3-1 Sequence Diagram of Triad 2: Publish/Request/Response Message Exchange Pattern

4.3.4 MEP Subscription

4.3.4.1 Description

The subscription message exchange pattern is used to provide a continuous data or product delivery service. The data or product can take the form of one or a series of messages. The steps for a subscription will typically be:

- Request/Response – REQ/RESP message pairs are used to request the data or product that is desired
- Publish - the MSG type messages are used by the publisher to distribute the specified data
- Request/Response - REQ/RESP message pairs are used to unsubscribe from the data

The subscription will remain active until it is cancelled by the subscriber. There is no restriction on the timing of the MSG messages, nor on the amount of messages. The subscription could result in one MSG, or a set of MSG messages. Additionally, the set of MSG messages could be periodically repeated.

The subscriber is free to cancel the subscription at any time; however, the provider can also terminate the subscription for its own reasons.

4.3.4.2 Usage

A subscriber requests a specific set of real-time mnemonic data values from a data provider. The provider will respond with a Response message indicating success or failure of the request. If the request was successful, mnemonic data values will be published. The subscription will remain open and on the next pass the requestor will again receive the specified real-time mnemonic data. The data will continue to be published for each pass until the requestor unsubscribes from the data. The subscriber can request the termination of the data values at any time.

In other scenarios, the subscriber may subscribe to previously known or predefined data sets already being published, rather than request specific data sets. Or, if enough information about a data set is made known and available, a consumer may simply read or ingest the data stream messages without even subscribing. There can be multiple subscribers for the data sets and products.

4.3.4.3 Sequence Diagram

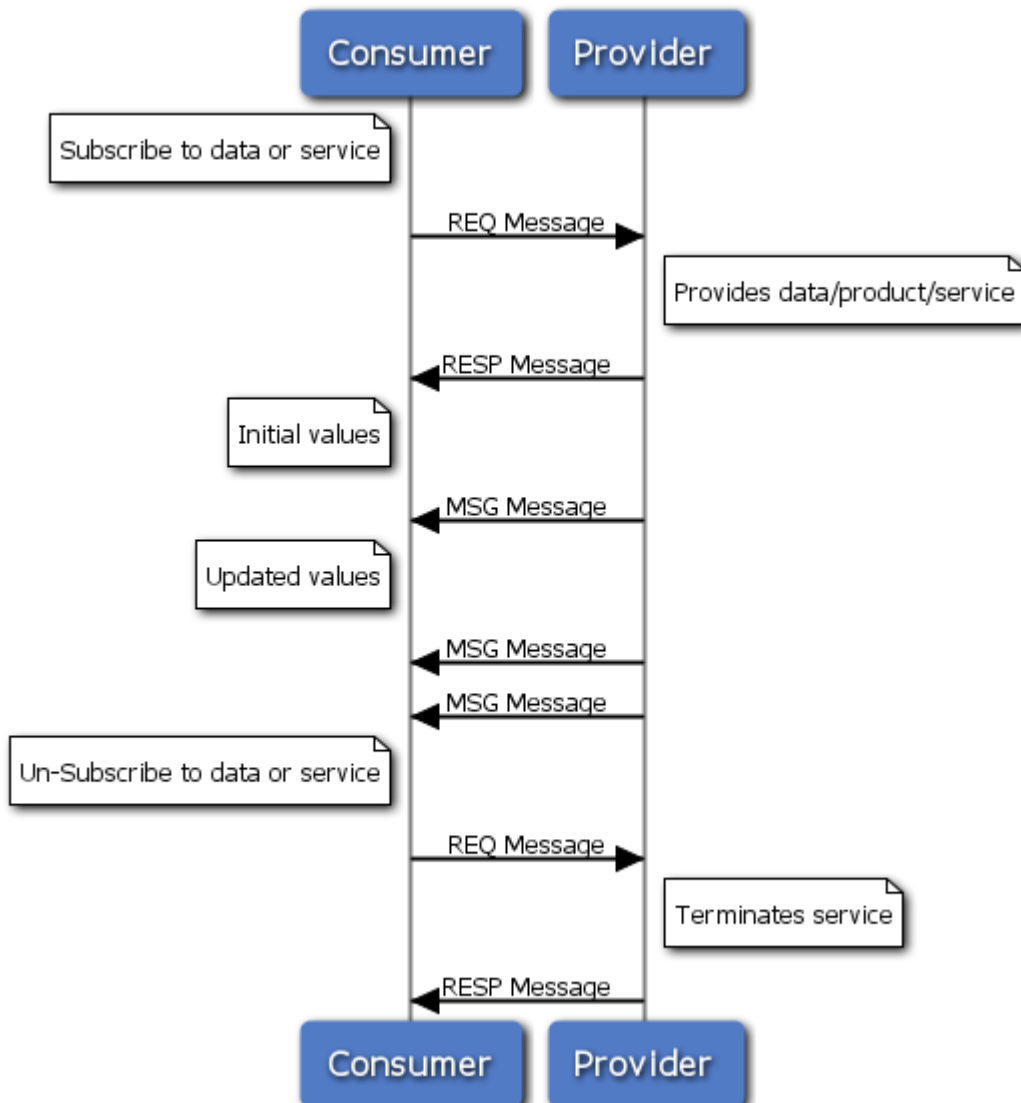


Figure 4.3.4.3-1 Sequence Diagram of Subscription Message Exchange Pattern

4.3.5 CCSDS Spacecraft Monitor and Control - Message Abstraction Layer

The CCSDS Spacecraft Monitor and Control (SM&C) Working Group is in the process of producing a number of specifications to define a set (framework) of standard services that will enable systems to be assembled from existing compliant components. It is expected that the “plug and configure” components will be able to be rapidly integrated into the system since their service interfaces have been standardized.

The SM&C Working Group has produced the following documents to facilitate the re-distribution of functionality between space and ground, systems, subsystems, and nodes. The openness and standardization of the architecture will promote re-use, visibility, availability, commonality, and ability to evolve for missions and ground systems. These evolving documents are:

- *Mission Operations Services Concept*
- *Mission Operations – Message Abstraction Layer*
- *Mission Operations – Reference Model*
- *Mission Operations – Common Object Model*
- *Spacecraft Monitor and Control – Common Services*
- *Spacecraft Monitor and Control – Core Services*
- *Asynchronous Message Service*

The Message Abstraction Layer (MAL) “defines a set of standard interaction patterns as an abstract interface that is used by the mission operations services.” By defining the standard patterns of interaction, messages can be exchanged for the operations of the services with confidence and reliability while focusing on the specifics of the operation.

The MAL has defined 6 interaction patterns. Each interaction pattern has a number of stages where messages are exchanged till the interaction pattern is completed. The interaction patterns and their respective stages are summarized in the following table.

Table 4-15. MAL Interaction Patterns with Associated Interaction Pattern Stages

Interaction Pattern Type		Message (Direction relative to provider)					
		Interaction Pattern Stage					
Name	Value	1	2	3	4	5	6
Send	1	Send (IN)					
Submit	2	Submit (IN)	ACK (OUT)				
Request	3	Request (IN)	Response (OUT)				
Invoke	4	Invoke (IN)	ACK (OUT)	Response (OUT)			
Progress	5	Progress (IN)	ACK (OUT)	Update (OUT)	Response (OUT)		
Publish-Subscribe (without broker)	6	Register (IN)	ACK (OUT)	Publish (IN & OUT to self)	Notify (OUT)	Deregister (IN)	ACK (OUT)

As seen from the table, the “Send” interaction pattern has only one state where the initial and only message is the “Send” message. The “Progress” interaction pattern has 4 stages to its pattern, using 4 different messages to complete the interaction. Please refer to the *Mission Operations Message Abstraction Layer, Draft Recommended Standard CCSDS 521.0-R-2, Red Book, September 2009* document for further detail on the MAL interaction patterns.

The GMSEC Architecture has defined four general (nine specific) interaction patterns similar to the MAL. Where the MAL uses a different message for each stage, GMSEC uses a combination of a message and a status code to identify the stage of the interaction. That is, the RESPONSE-STATUS field of the Response message indicates the stage of the interaction. It can signify an ACK, a Response, an error message, and even a Final Message of a series. For error messages, the RESPONSE-STATUS field of a Response message will return error codes to perform the same function as MAL error messages. The values of the RESPONSE-STATUS field are provided in Table 4-14. Response Status Substructure. For the Publish-Subscribe pattern, GMSEC generally assumes the broker and provider are the same. That is, there is no intervening broker. GMSEC consumer components perform publish-subscribe interactions directly with the provider of the service. The following table compares the SM&C MAL interaction patterns with their respective stages to the corresponding GMSEC protocol.

Table 4-16. Comparison of SM&C and GMSEC Interaction Patterns

SM&C	No.	Interaction Pattern Name	Stage						
GMSEC	No.	Interaction Pattern Name	1	2	3	4	5	6	
SM&C	1	Send	Send						
GMSEC	1	Publish	Publish						
SM&C	2	Submit	Submit						
GMSEC	2.1	Request / ACK	Request	ACK	Response (Acknowledge)				
SM&C	3	Request	Request						
GMSEC	2.2	Request / Response	Request	Response	Response (Status)				
SM&C	4	Invoke	Invoke						
GMSEC	2.3	Request/ACK/Response	Request	ACK	Response	Response (Status)			
SM&C	5	Progress	Progress						
GMSEC	2.4	Request/ ACK/ Interim Status/ Response	Request	ACK	Update	Response			
				Response (Acknowledge)	Response (Working)	Response (Status)			
SM&C		None							
GMSEC	2.5	Request / Interim Status / Response	Request	Response (Working) ...	Response (Status)				
SM&C		None							
GMSEC	3.1	Request/ Response/ Publish (Triad 1)	Request	Response (Status)	Publish				
SM&C		None							
GMSEC	3.2	Publish/ Request/ Response (Triad 2)	Publish	Request	Response (Status)				
SM&C	6	Publish-Subscribe with Broker	Register	ACK	Publish	Notify ...	Deregister	ACK	
GMSEC		None							
SM&C	6	Publish-Subscribe without Broker	Register	ACK	Publish (to self)	Notify ...	Deregister	ACK	
GMSEC	4	Subscription	Request	Response	NA	Publish ...	Request	Response	

4.4 Product Messages

4.4.1 GMSEC and Component Handling

4.4.1.1 Background and Scope

The GMSEC Information Bus architecture provides a number of options for product generators and producers to make their products available, as well as for product consumers to access and receive these products. Options are provided for producers that perform product generation automatically, by request, in various formats and sizes, and that also have a variety of means for distribution. GMSEC seeks to facilitate the distribution of these products. It does not intend to direct how, when, and under what circumstances products are created. The design, format, frequency, and circumstances of the product generation are left to the producers and their clients. Also, in that products are generally formed as files to be transferred between locations, please see Section 1.3.1 File Transfers for a discussion on this topic.

4.4.1.2 Product Generation and Distribution Factors

The GMSEC API, middleware, and message structure provides a flexible number of transport mechanisms and protocols to distribute products component-to-component. These distribution options are themselves guided and bounded by other factors determined by mission operations concepts and procedures, protocols inherent within the product generators, and capabilities of the underlying middleware and operating system platforms. Some of these factors include:

Operations Concepts:

- When is a product generated? When is a product distributed?
- Who are the expected users/consumers of products?
- To where are the products to be distributed?
- Do the producer and/or client determine when and where a product is to be distributed?

Authorization factors:

- Where may a product be distributed?
- What components or users have access to a product or distribution area?
- Which clients and users may request product distribution?

Protocol factors:

- How is a product requested?
- How can a product be distributed?
- In what formats are the products to be generated or presented?

4.4.1.3 GMSEC Product Message Summary

GMSEC has defined the following messages to facilitate the needs of product producers and consumers.

- **Product Request Message** – used to request a product
- **Product Response Message** – used to return status of the request, and optionally, to provide the product
- **Product Message** – used to
 - 1) Announce the **availability** of a generated product,
 - 2) Announce a product is **accessible** by providing the location with a Uniform Resource Identifier (URI), or
 - 3) Provide the Product in the message as an **attachment**.

Table 4-17. Uses of the Product Message

Usage	User Required Action
Available	Must request the product
Accessible	Use the URI to get the product
Attachment	Extract the product from the message

The Product Message is published in one of two circumstances.

1. After the exchange of the Product Request and Product Response Messages
2. Unsolicited

The Product Response Message and the Product Message are used to distribute products. These messages are used for a single product that may contain a multiple number of files. Generally, the contents of the different messages are as follows:

Product Request Message

- Requests the distribution of product(s) (that might require the producer to generate)
- Specifies attributes to describe the requested product(s)
- Provides information to direct the means of distribution and/or target the distribution location
- Optionally, includes precursor products (files) that are used to generate the requested product

Product Response Message

- Return Status of the Product Request
- Optionally,
 - Contains the actual product or product location information
 - Contains product attributes

Product Message

- Contains the actual product or product location information
- Contains product attributes

The GMSEC Product Request Message effectively requests the distribution of a product. It may incidentally require the generation of that product by the producer if it does not already exist. The GMSEC Product Response Message and the Product Message incorporate a framework to identify the number of files per product. The messages also allow for determining the location of the distribution. The requestor could specify the location or allow the producer to specify the location. Of course, these locations are dependent on the granted access and authorization of components to these designated locations.

Following is a summary of the ways and means product producers operate that GMSEC will accommodate within its architecture.

4.4.2 Product Characteristics

4.4.2.1 Production

4.4.2.1.1 Methods of Production

Automatic production

- Some products are generated automatically. This can occur on a periodic basis, upon the availability and reception of data, upon the occurrence of an event, etc.
- Examples: orbital event files, orbit determination calculations, schedules, telemetry data derivatives, public consumables

Production by request

- Production by request is effectively distribution by request. It is possible that the product may already exist, or that it may need to be generated. It may be a standard product or one specifically requested by the client that will be customized. In any case, the producer will end up distributing the product, as requested.
- Examples: telemetry data extractions, plots, custom statistics, etc.

4.4.2.1.2 GMSEC Facilitation

Automatic Production

- GMSEC has only indirect involvement in the production. Inputs (files, data) to the product generator are made available or accessible through the use of the GMSEC Information Bus.
- The GMSEC Product Message is used to announce the generated product or to automatically publish the product without it being specifically requested. The announcement of the product could merely indicate the

product is available and can be requested, or the announcement might provide a location for clients to access the product. Other product information is also available in the Product Message such as product type, format, version, date, file name, etc.

Production by Request:

- The GMSEC Product Request and Response Messages are used to request the generation and distribution of a product from a producer. The requestor has options to specify how the product is to be distributed.

By what message:

- Product Response Message, or
- Product Message

In what form:

- By reference
- Included in the message

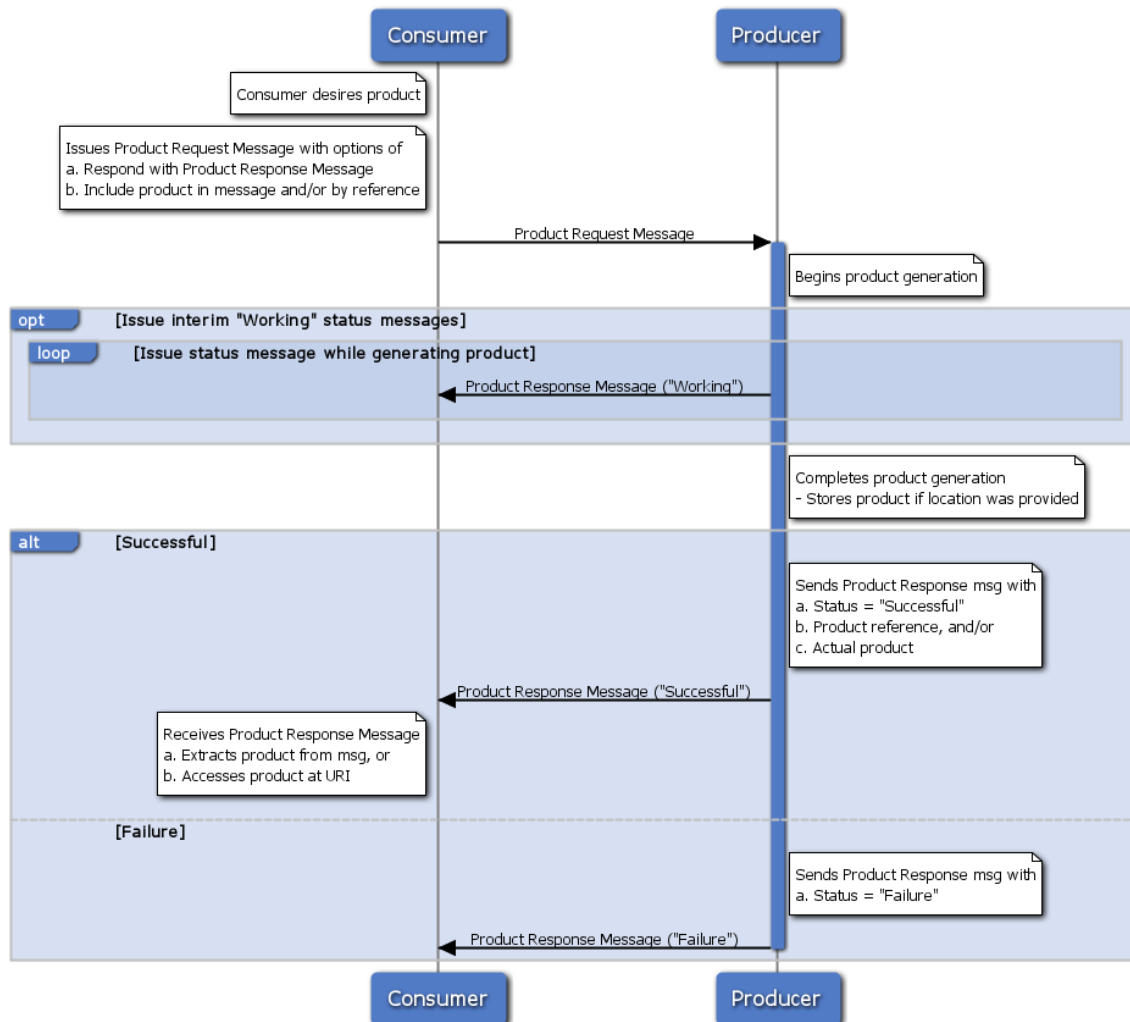
Other ancillary product information can also be specified in the request. If the delivery of the product is by the Product Message the requestor can subscribe to the subject of the Product Message that will later be published with the included product.

The Product Request and Product Response messages will follow the protocol established for the other request/response interactions. This is further described in Section 4.2 GMSEC Messages: Their Characteristics and Interactions.

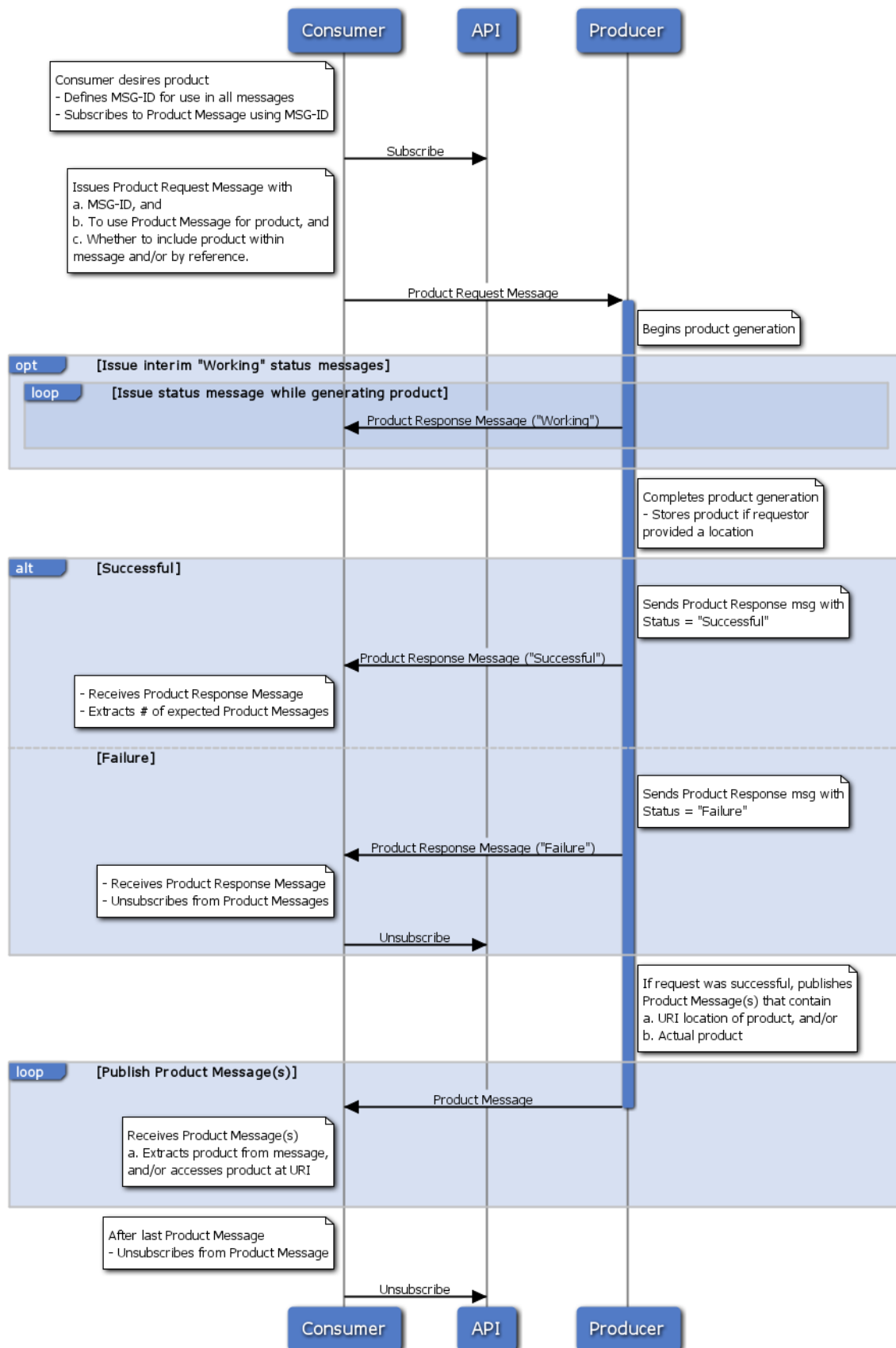
Four example scenarios have been provided.

- Scenario 1 - uses the Product Request and Product Response messages to request and distribute a product
- Scenario 2 - uses the Product Message in addition to the Product Request and Response messages
- Scenario 3 - shows a client receiving a product automatically generated and distributed by a producer
- Scenario 4 – shows a client receiving a product announcement in the Product Message, and proceeds with Scenario 1 to request the product

Scenario 1: Client requests a product using only the Product Request and Product Response Messages. Product (or product reference) is supplied in Response Message.

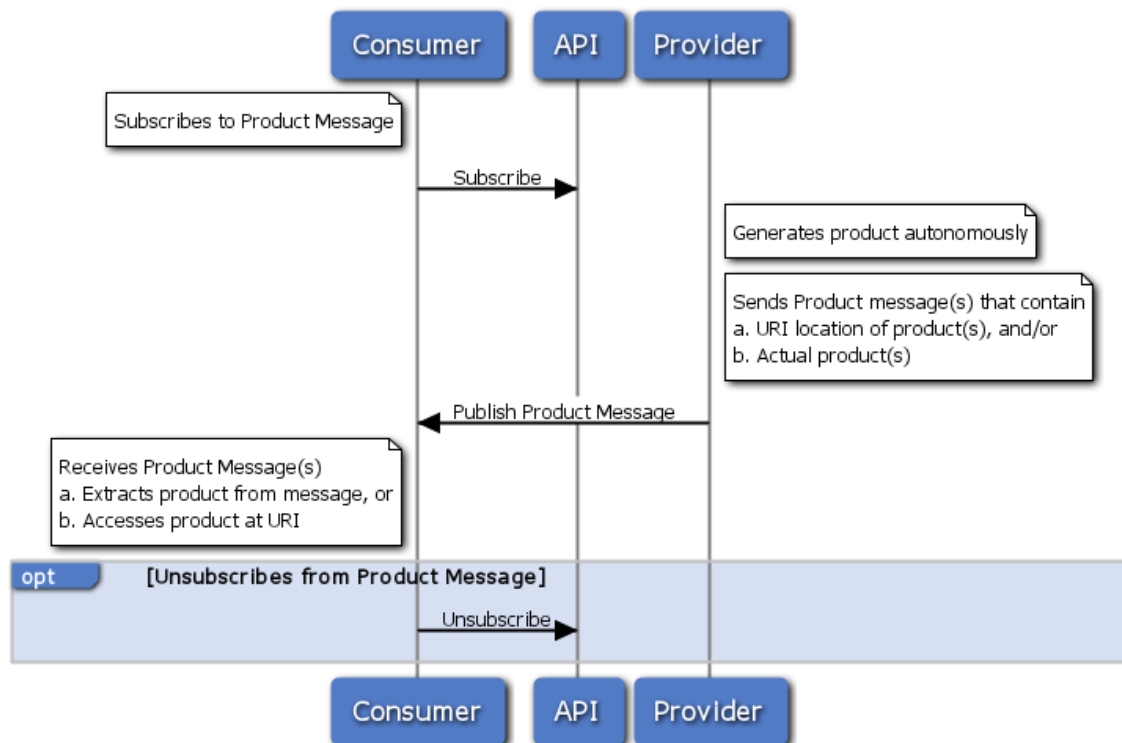


Scenario 2: Client requests a product using the Product Request, Product Response, and Product Messages. Product (or its reference) is supplied in the Product Message following the Request/Response exchange. The sequence diagram follows on the next page.

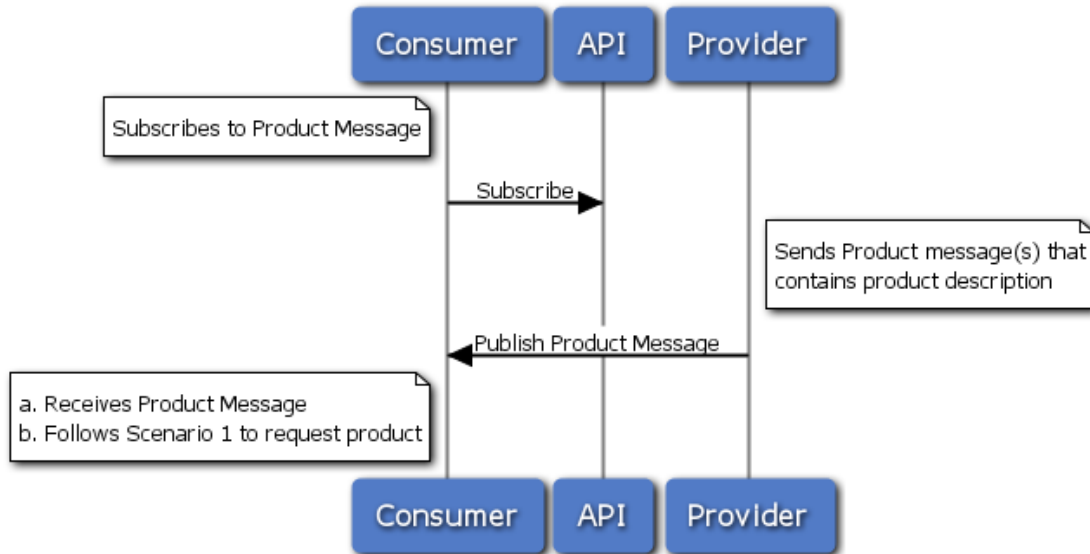


Scenario 2a: It may not be known how many Product Messages will be produced and sent. In this case the field PROD-MSG-TO-SEND can be set to “-1” and then send a Product Response Message at the conclusion of the stream of Product Messages. The RESPONSE-STATUS field would contain a “FINAL-MESSAGE” status code.

Scenario 3: Client subscribes to products automatically produced and distributed by Producer using Product Message. The client or consumer will need to know the details of the message subject elements of the Product Message. No Request/Response interaction is required.



Scenario 4: Client subscribes to Product Message awaiting announcement from provider that a desired product is now available (or could be generated). Client then issues Product Request message and receives product (or its reference) in Product Response.



4.4.2.2 Content and Format

4.4.2.2.1 Forms of Products

Products are considered to be in a file format. Depending on the product, the capabilities of the product generator, and the needs of the clients, products may take a number of different formats. These could include:

- Data (text, pdf, spreadsheet, CCSDS, OMG, XML, ...)
- Images
 - Video/movie
 - Snapshot
- Audio
- Other

One product may consist of one file, or many files. The files may be in the same or different formats. The products may be organized in a hierarchical fashion as follows:

- Carton or case, consisting of one or more products
 - Product, consisting of one or more files
 - File, in a variety of formats

The GMSEC Product Message and Product Response Messages are used for a single product, that is, one product per message. The Product Response and Product Messages allow for multiple files per product.

4.4.2.2.2 Product Attributes or Properties

Attributes Desired to Accompany a Product

Producer

(The next two attributes could act together as a unique ID).

Product Name (in the form of a type and subtype)

Time of Creation

Version

Format

Size (in kilobytes)

Location information such as file name and specification

Number of files

4.4.2.2.3 GMSEC Facilitation

The GMSEC Product Request and Response Messages will allow for the distribution of a single product of one or more files. The Response Message will be used to provide status on the validity of the request, and optionally, the actual product files. The GMSEC Product Message is used to send the product with all

its attributes, or simply provide a pointer to the accessible location of the product(s).

GMSEC, the product producer, product consumer, and/or the middleware will need to limit or be aware of the size of the product(s) being requested and transported over the GMSEC Information Bus. The participants in the product generation and transfer may be limited in the following ways:

Consumer/Requestor:

The consumer may request a product be stored in a location that has resource limitations. The consumer can specify in the request that a limit for the product size not be exceeded.

Producer/Responder:

Likewise, a producer may be limited in its storage capacity.

Middleware

Middleware are bounded by the size of messages, buffers, memory, and other resources that could prevent the transfer of a large number of large items. Some middleware may provide for (lower) prioritizing of large file or message transfers.

4.4.2.3 Distribution

4.4.2.3.1 Product Generation and Distribution

Product distribution is reflective of the concepts behind the product generation. The table below summarizes the possible relationships.

Table 4-18. Relationship of Product Generation and Distribution

PRODUCT GENERATION	PRODUCT DISTRIBUTION	
	Automatic	By Request
Automatic	Products are automatically generated and immediately distributed.	Products are automatically generated and available, but only distributed upon request.
By Request	NA	Products are generated upon request and immediately distributed.

Examples of products automatically distributed include: orbital events, orbit determinations, schedules, telemetry derivatives, and public consumables.

Examples of products generated and distributed only upon request include: telemetry data point extractions, plots, and customized statistics.

4.4.2.3.2 GMSEC Facilitation

Mission operations concepts and procedures determine the final location of product distribution, not GMSEC. GMSEC facilitates message distribution among components on the GMSEC Information Bus, but not the final location where the product will be stored. Product distribution may occur in the following ways:

- Point-to-point, that is from one component (producer) to another (consumer) with no regard for a final storage location, if at all
- To the requestor's (client/consumer) specified designation (using a URI)
- To the producer's own server repository to be subsequently copied by the client
 - This area must be accessible by the clients
- To a central or mission repository or archive
- To a publicly available repository or library

The producer or the consumer of the product could initiate product distribution.

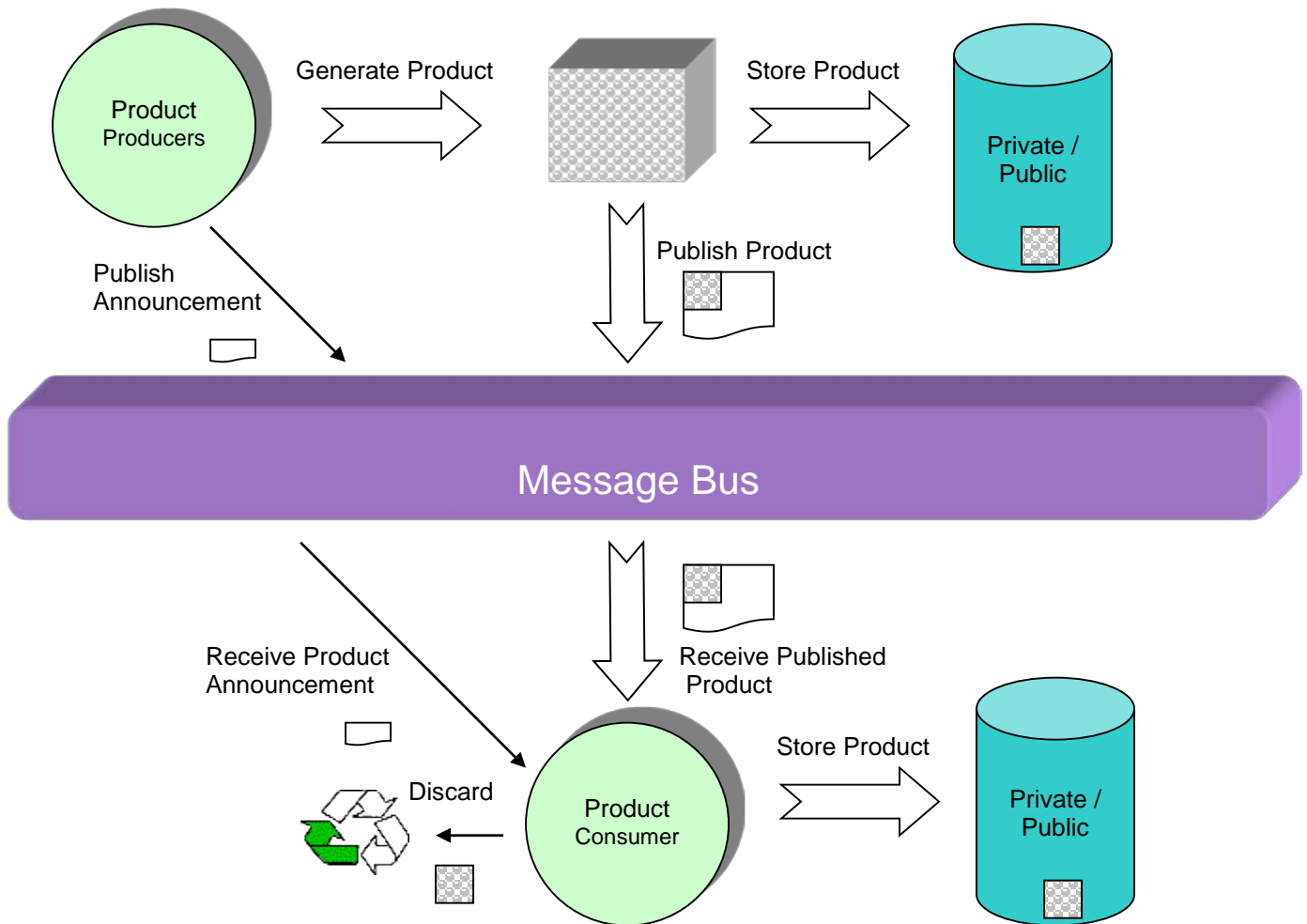


Figure 4.4.2.3.2-1 Product Distribution Options

Table 4-19. Product Generation and Distribution Scenarios

Generation and Distribution Scenarios				
	Product		GMSEC Transport Mechanisms	
	Generation	Distribution	Producer	Consumer
1	Automatic	Immediate & Automatic	<ul style="list-style-type: none"> - Sends Product Message for notification of generation only (no product included), or - Sends Product Message with product included for general distribution 	<ul style="list-style-type: none"> - Use URI in message to access product, - Or, Extract product from message
2	Automatic	Later By Request	1. Sends Product Message with notification of generation only 3. Sends Product Response Message AND/OR 4. Sends Product Message with product included or URI location	2. Sends Request Message to request distribution
3	By Request	Immediate By Request upon generation of product	2. Sends Response Message 3. Generates product 4. Sends Product Message with product included or URI location	1. Sends Request Message to request distribution

Components also provide different levels of services. For example, a product generator might also provide an advertised directory of products, either those that exist in a repository, or those that can be generated. This information could be disseminated in a variety of ways including as a service, as a request/response message exchange, by separately viewing an accessible library or repository, publishing log messages when products are available (with product specific information), and so on. The component might also provide a means to window shop the available products. The user might be able to select products, put them in a shopping cart, and then request distribution (ignoring the monetary considerations).

4.4.3 Product Message Examples

The following diagrams depict a user request for a data plot, and how some of the GMSEC messages might be used.

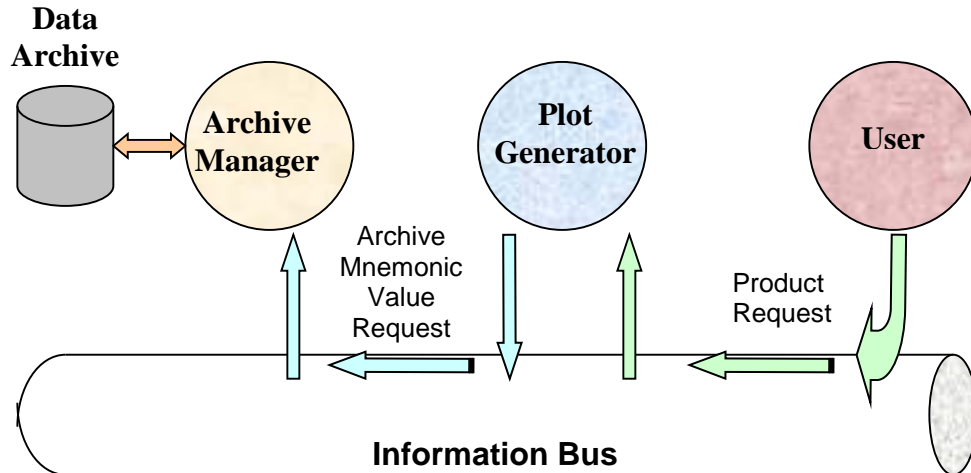


Figure 4.4.3-1 Requests for Data Plot Using GMSEC Messages

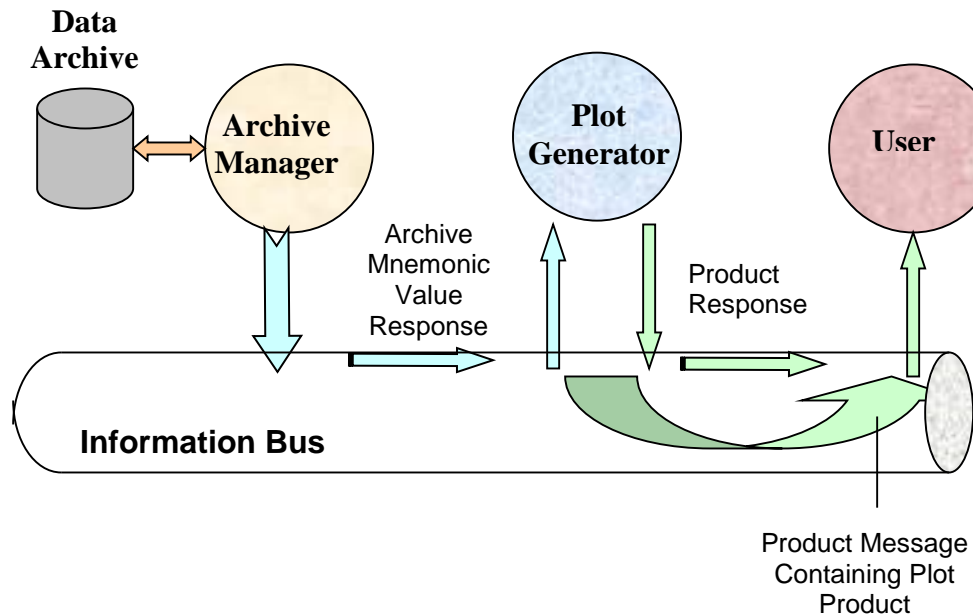


Figure 4.4.3-2 Responses to Data Plot Request Using GMSEC Messages

4.5 Categorization

4.5.1 Message Categories and Message Overview

The initial scope and approach of the GMSEC architecture was intended to work within the existing spacecraft and ground system development culture at NASA, existing/emerging COTS and GOTS products, the limitations of available flight hardware, and also within current budgets. The initial focus and implementation of these messages has primarily been for the ground systems. As the implementation of this architecture migrates to spacecraft systems, existing messages may be adapted and new messages may be defined. Furthermore, the GMSEC Architecture need not be constrained to NASA only, but has proven to be applicable to other cultures, agencies, and enterprises.

As the implementation of GMSEC progressed, many items have been defined and categorized. Categorization has been applied to messages, components, products, and other areas. Categorization has significantly enabled the exchange of messages while decoupling the interdependence between components. Some of this categorization has included:

Messages

- Type
 - Communication mechanisms
 - Request/Response, Publish/Subscribe
- Subtype
 - Identifier (as a textual abbreviation) of the message function

Components

- Location
 - Facility, Node
- Identifier
 - Component name, subcomponent, sub-subcomponent, process ID, ...
- Function
 - Class

Products

- Type, subtype, sub-subtype, ...

The following table organizes the messages by function (column). Specific message definitions are provided in Section 5 GMSEC Standard Messages and grouped the same way as in the table that follows.

Table 4-20. GMSEC Message Functional Grouping

Control and Monitor Level	Data Level	Product and Service Level
<ul style="list-style-type: none"> Log Message 	<ul style="list-style-type: none"> Telemetry Message Tracking Data Message 	<ul style="list-style-type: none"> Product Request Product Response Product Message
<ul style="list-style-type: none"> Archive Message Retrieval Request Archive Message Retrieval Response 	<ul style="list-style-type: none"> Replay Telemetry Request Replay Telemetry Response 	<ul style="list-style-type: none"> Attitude Parameter Message Attitude Ephemeris Message Orbit Parameter Message Orbit Mean-Elements Message Orbit Ephemeris Message
<ul style="list-style-type: none"> Directive Request Directive Response 	<ul style="list-style-type: none"> Mnemonic Value Request Mnemonic Value Response Mnemonic Value Data Message 	<ul style="list-style-type: none"> Simple Service Request Simple Service Response
<ul style="list-style-type: none"> Component-to-Component Transfer Message <ul style="list-style-type: none"> Configuration Control Device Heartbeat Resource 	<ul style="list-style-type: none"> Archive Mnemonic Value Request Archive Mnemonic Value Response Archive Mnemonic Value Data Message 	
	<ul style="list-style-type: none"> Database Attributes Request Database Attributes Response 	
	<ul style="list-style-type: none"> Command Request Command Response 	

A brief description of the function of the messages follows.

4.5.1.1 Control and Monitor

In general, Control and Monitor messages are used to

- Convey status information about a component
- Control or direct a component to action

There are four kinds of Control and Monitor messages. They are:

- **Log Message**
Used to notify or “log” the occurrence of an event, or pass along data or state information
- **Archive Messages**
Used to retrieve messages from an archive. It allows for the retrieval of any message that was previously archived.
- **Directive Messages**
Directive messages are used to direct another component to action. The Directive message provides a framework to pass along an operations language command string that the receiving component must be able to parse and execute. These messages generally are of interest to the operations personnel and control the operation of the system.
- **Component-to-Component Transfer**
These messages are similar in nature to the Directive messages but occur at a lower level than the Directive messages. They are generally not of interest at the system operations level. Components use these messages to convey information and also to direct / control one another.

4.5.1.2 Data Messages

Data messages are used to pass along

- Real-time data
 - Data streams (packets, frames)
 - Data points
- Historical data
 - Data streams (packets, frames)
 - Data points
- Data attributes

The Data Messages are:

- **Telemetry and Command Packets and Frames**
These messages are real-time data streams.
- **Replay Telemetry**
Generally, these messages are used to send streams of data that have been retrieved out of an archive of previously recorded data streams. However, the message definition can be used to request the publication of a past, present, or future telemetry data stream.
- **Mnemonic Value Messages**
The Mnemonic Value messages send specific data point values in real-time.
- **Archive Mnemonic Value Messages**
The Archive Mnemonic Value messages send specific data point values that have been retrieved from an archive.
- **Database Attributes**
These messages are used to request and receive attribute information about mnemonics. They are complementary to the Mnemonic Value and Archive Mnemonic Value messages.

4.5.1.3 Product and Service Messages

Product Messages are used to convey information about products that have been or could be generated. The messages can be used to

- Announce a product is available and can be requested from the producer
- Announce a product is accessible and can be retrieved at a specified location
- Transfer a product from one component to another

Note that the product messages are not concerned with the storage of a product (by the producer or consumer). They merely move the product (or product information) between components.

Navigation Data Messages are a kind of Product Message.

Simple Service messages are used to request a service offered by another component. The Simple Service messages utilize a Request/Response message exchange pattern to provide a framework for the service exchange. The consumer will pass along the service and operation identifier along with any required parameters to the producer. The producer will respond with the status and any necessary data.

4.5.2 Component Categories

The following table organizes the software into broad classes of functionality and further delineates the classes into subclasses. 6 GMSEC Services provides an example of how the classes or subsystems of a ground system might interact with GMSEC standard messages.

Table 4-21. Software Class and Subclass Categories

CLASS	Class Abbr.	Subclass	Subclass Abbr.
Archive and Assessment	AAA	Assessment Archive Plotting, Trending and Analysis	AST ARC PTA
Automation	AUTO	Paging Rule-Action	PAGE RULE
Flight Dynamics	FD	Orbit Determination Navigation and Control Ephemeris	OD NAC EPH
Modeling, Simulation, and Front End Processors	MAS	Analysis Front End Processor Simulator	ANL FEP SIM
Planning and Scheduling	PAS	Maneuver Planning Scheduling	MAN SCH
Script Control	SCRIPT		
Security	SEC	Verify Encryption	VER CRYPT
System	SYS	Operating Systems COTS GOTS Monitor Configuration Control and Management	OS COTS GOTS MON CFG
Telemetry and Command	TAC	Telemetry Command	TLM CMD

(Not a complete listing)

4.5.3 Product Categories

This table uses the same Class categories and Class abbreviations (as seen in Table 4-28. Software Class and Subclass Categories) to breakdown and classify the products in a hierarchical manner.

Table 4-22. Product Categories

Product Types (ME3)	Abbr.	Product Subtype (ME4)	Abbr.	Product Subtype2 (ME5)	Abbr.	Product Subtype3 (ME6)	Abbr.
Archive and Assessment	AAA	Data	DATA				
		Message	MSG				
		Plot/Graph	PLOT				
		Report	RPT				
		Statistics	STAT				
Automation	AUTO	Decision Making	DM				
		Expert	EXP				
		Paging	PAGE				
Flight Dynamics	FD	Attitude	ATT				
		Environmental/Celestial	EC				
		Instrument	INST				
		Maneuver	MAN				
		Memory	MEM				
Flight Dynamics	FD	Orbit	ORBIT	Ephemeris	EPHEM	Binary	BIN
						FreeFlyer	FF
						CCSDS Orbit Ephemeris Message	OEM
						Satellite Tool Kit	STK
						Special Perturbations	SP

Product Types (ME3)	Abbr.	Product Subtype (ME4)	Abbr.	Product Subtype2 (ME5)	Abbr.	Product Subtype3 (ME6)	Abbr.
Flight Dynamics	FD	Orbit	ORBIT	State	STATE	Extended Precision Vector	EPV
						GPS Navigation Data	GPSNAV
						Improved Interrange Vector	IIRV
						CCSDS Orbit Parameter Message	OPM
						Orbital Parameter Reports	OPR
						Two Line Mean Elements	TLE
						Vehicle Attitude	VEHATT
Flight Dynamics	FD	Orbit	ORBIT	Station Contact	STA	Az/EI Tables	AZEL
						Ground Site Location	GSL
						Line Summary	LS
						Predicted Site Acquisition Table	PSAT
						Site Views	SV
						STDN Summary Predictions	STDNSP
						Tracking Data	TD
						Vehicle Visibility Check	VVC
Flight Dynamics	FD	Orbit	ORBIT	Instrument	INST	High Gain Antenna Predictions	HGAP
						Science Field of View Predictions	SFVP
						Sensor Interference Predictions	SIP
						User Antenna View	UAV

Product Types (ME3)	Abbr.	Product Subtype (ME4)	Abbr.	Product Subtype2 (ME5)	Abbr.	Product Subtype3 (ME6)	Abbr.
Flight Dynamics	FD	Orbit	ORBIT	Orbit/Object Relationships	OOR	Orbital Events	EV
						Shadow Times	ST
						Solar Beta Angle	SBA
						Sun/Earth Relationships	SER
Modeling and Simulation	MAS	Data Files	DATA				
		Data Streams	STREAM				
		Messages	MSG				
Planning and Scheduling	PAS	Schedule	SCH	Activity Schedule	ACT		
				Contact Schedule	CON	Air Force Satellite Control Network	AFSCN
						Deep Space Network	DSN
						Ground Network	GN
						Space Network	SN
						Universal Space Network	USN
Planning and Scheduling	PAS	Spacecraft	SC	Command	CMD	Command Sequence File	CSF
						Block Commands	BC
		Unmanned Aerial Vehicles	UAV				
Scripting Control	SCRIPT						

Product Types (ME3)	Abbr.	Product Subtype (ME4)	Abbr.	Product Subtype2 (ME5)	Abbr.	Product Subtype3 (ME6)	Abbr.
Telemetry and Command	TAC	Data Values	DATA	Selected	SET	ID of selected subset	nnn
		Database	DB	Limit Sets	LIMITS		
				Text Conversion	TEXT		
				Calibration Curves	CAL		
				Mnemonics Short Description	MNSHORT		
				Mnemonics Long Description	MNLONG		
				Mnemonics All	MNALL		
Telemetry and Command	TAC	Level-0	RAW				
Telemetry and Command	TAC	Memory Dumps	MEM				
Telemetry and Command	TAC	Products Derived from a pass/contact	PASS	Pass Description	PD		
	TAC			Frames Lost	FL		
				Pass Summary Report	PSR		
		Spacecraft Tables	TBL				

Section 5 GMSEC Standard Messages

5.1 Message History

5.1.1 Message Evolution Pages

A mapping of the history of the messages to the Interface Specification is given in the following tables. **The column titled “Last Defined or Modified Version” highlights changes from the previous version.** If the message is new or fields have been added, the message is considered “modified”. If fields in the message have only been rearranged, the message is considered unchanged.

Upon release of this document, missions, components, users, and other implementation architects should review these tables to see if a message has evolved to a higher version than currently supported. A mission or authorizing body may need to determine if the new version and/or previous versions of a message shall be supported. That is, what version (s) of the Interface Specification will be supported? Those responsible for the components may want to be able to support multiple versions of a message in order for the component to be easily deployed in a variety of circumstances.

For components that support an evolving message, or components that support multiple missions, it may be necessary to support multiple versions of a message. The following component processing logic might be employed to provide the best service while the GMSEC Architecture and implementation evolves into a true, tested, and stable standard.

A component will need to pre-determine what messages and what version(s) of a message it will support. This may be a single version of a message or multiple versions of a message. On the input (receiving) side, a component should check the version number in the CONTENT-VERSION field (Required) of the message. If the version of the incoming message is not consistent with its pre-determined list of versions it will process, it should return a response with a RESPONSE-STATUS code of "Invalid Request" (5) and/or issue a Log message. In order to be backward compatible, a component must decide if it will process one or more versions of a message. On the output (sending) side, if a response message is required for a received message, the component should send the complementary version. That is, if the request message has evolved to a 1.3 version but the response message has remained at a 1.2 version, then these are the versions of the messages that should be supported.

5.1.2 Message Definition History

Table 5-1. GMSEC Message Definitions History, 1 of 4

Message Name	Changes since last revision	Last Defined or Modified Version	Document Section	Page Number
GMSEC Information Bus Header		2010	4.1.2 GMSEC Information Bus Header	40
Control and Monitor Messages			5.2 Control and Monitor Level Messages	
Log	Replace MSG-ID with UNIQUE-ID	2016	5.2.1.1 Log Message	105
Archive Message Retrieval Request	Replace MSG-ID with UNIQUE-ID and signed with unsigned fields	2016	5.2.1.2.1 Archive Message Retrieval Request	116
Archive Message Retrieval Response	Replace MSG-ID with UNIQUE-ID and signed with unsigned fields	201^	5.2.1.2.2 Archive Message Retrieval Response	124
Directive Request	Replace MSG-ID with UNIQUE-ID	2016	5.2.2.1.1 Directive Request Message	132
Directive Response	Replace MSG-ID with UNIQUE-ID	2016	5.2.2.1.2 Directive Response Message	135
C2CX Configuration	Replace MSG-ID with UNIQUE-ID and signed with unsigned fields	2016	5.2.2.2.2.1 Component-To-Component Transfer Configuration Status Message	141
C2CX Control	Replace MSG-ID with UNIQUE-ID	2016	5.2.2.2.2.2 Component-To-Component Transfer Control Message	144
C2CX Device	Replace MSG-ID with UNIQUE-ID and signed with unsigned fields	2016	5.2.2.2.2.3 Component-To-Component Transfer Device Message	146
C2CX Heartbeat	Replace MSG-ID with UNIQUE-ID and signed with unsigned fields, add optional COMPONENT-INFO-DETAILS field	2016	5.2.2.2.2.4 Component-To-Component Transfer Heartbeat Message	148
C2CX Resource	Replace MSG-ID with UNIQUE-ID and signed with unsigned fields, add information for Main Memory	2016	5.2.2.2.2.5 Component-To-Component Transfer Resource Message	151

Table 5-2. GMSEC Message Definitions History, 2 of 4

Message Name	Change Since Last Revision	Last Defined or Modified Version	Document Section	Page Number
Data Messages			5.3 Data Level Messages	
Telemetry CCSDS Packet	Replace MSG-ID with UNIQUE-ID, require message to contain packet, remove requirement for length enumeration	2016	5.3.1.1.2.1 Telemetry Message Contents for CCSDS Packet	160
Telemetry CCSDS Frame	Replace MSG-ID with UNIQUE-ID and signed with unsigned fields	2016	5.3.1.1.2.2 Telemetry Message Contents for CCSDS Frame	161
Telemetry TDM Frame	Replace MSG-ID with UNIQUE-ID and signed with unsigned fields	2016	5.3.1.1.2.3 Telemetry Message Contents for TDM Data	163
Processed Telemetry	Replace MSG-ID with UNIQUE-ID and signed with unsigned fields	2016	5.3.1.1.2.4 Telemetry Message Contents for Processed Telemetry Frame	164
Replay Telemetry Request	Replace MSG-ID with UNIQUE-ID and signed with unsigned fields	2016	5.3.1.2.1 Replay Telemetry Data Request Message	170
Replay Telemetry Response	Replace MSG-ID with UNIQUE-ID	2016	5.3.1.2.2 Replay Telemetry Data Response Message	175
Mnemonic Value Request	Replace MSG-ID with UNIQUE-ID and signed with unsigned fields	2016	5.3.2.1.1 Mnemonic Value Request Message	180
Mnemonic Value Response	Replace MSG-ID with UNIQUE-ID and signed with unsigned fields	2016	5.3.2.1.2 Mnemonic Value Response Message	187
Mnemonic Value Data Message	Replace MSG-ID with UNIQUE-ID and signed with unsigned fields	2016	5.3.2.1.3 Mnemonic Value Data Message	193
Archive Mnemonic Value Request	Replace MSG-ID with UNIQUE-ID and signed with unsigned fields	2016	5.3.2.2.1 Archive Mnemonic Value Request Message	202
Archive Mnemonic Value Response	Replace MSG-ID with UNIQUE-ID and signed with unsigned fields	2016	5.3.2.2.2 Archive Mnemonic Value Response Message	210
Archive Mnemonic Value Data Message	Replace MSG-ID with UNIQUE-ID and signed with unsigned fields	2016	5.3.2.2.3 Archive Mnemonic Value Data Message	215
Database Attributes Request	Replace MSG-ID with UNIQUE-ID and signed with unsigned fields	2016	5.3.2.3.1 Database Attributes Request Message	225
Database Attributes Response Limit Sets	Replace MSG-ID with UNIQUE-ID and signed with unsigned fields	2016	5.3.2.3.2 Database Attributes Response Message	230
Database Attributes Response	Replace MSG-ID with UNIQUE-ID and signed	2016	Table 5-111. Database Attributes Response	233

Text Conversion	with unsigned fields		Message Contents for Text Conversion	
Database Attributes Response Calibration Curve	Replace MSG-ID with UNIQUE-ID and signed with unsigned fields	2016	Table 5-112. Database Attributes Response Message Contents for Calibration Curve	234
Database Attributes Response Short Description	Replace MSG-ID with UNIQUE-ID and signed with unsigned fields	2016	Table 5-113. Database Attributes Response Message Contents for Short Description	235
Database Attributes Response Long Description	Replace MSG-ID with UNIQUE-ID and signed with unsigned fields	2016	Table 5-114. Database Attributes Response Message Contents for Long Description	236
Database Attributes Response List of All Mnemonics	Replace MSG-ID with UNIQUE-ID and signed with unsigned fields	2016	Table 5-115. Database Attributes Response Message Contents for List of All Mnemonics	237
Command Request	Replace MSG-ID with UNIQUE-ID and signed with unsigned fields	2016	5.3.3.1 Command Request Message	241
Command Response	Replace MSG-ID with UNIQUE-ID	2016	5.3.3.2 Command Response Message	245

Table 5-3. GMSEC Message Definitions History, 3 of 4

Message Name	Change Since Last Revision	Last Defined or Modified Version	Document Section	Page Number
Products & Services Messages			5.4 Products and Services Level Messages	
Product Request	Replace MSG-ID with UNIQUE-ID and signed with unsigned fields	2016	5.4.1.1 Product Request Message	250
Product Response	Replace MSG-ID with UNIQUE-ID and signed with unsigned fields	2016	5.4.1.2 Product Response Message	253
Product Message	Replace MSG-ID with UNIQUE-ID and signed with unsigned fields	2016	5.4.1.3 Product Message	256
Simple Service Request	Replace MSG-ID with UNIQUE-ID and signed with unsigned fields	2016	5.4.2.1 Simple Service Request Message	266
Simple Service Response	Replace MSG-ID with UNIQUE-ID	2016	5.4.2.2 Simple Service Response Message	270

Table 5-4. GMSEC Message Definitions History, 4 of 4

Message Name	Change Since Last Revision	Last Defined or Modified Version	Document Section	Page Number
Navigation Data Messages			5.5 Navigation Data Messages	
Attitude Parameter Message	Replace MSG-ID with UNIQUE-ID, remove LENGTH descriptor for DATA	2016	5.5.1.1 Attitude Parameter Message (APM) Contents	279
Attitude Ephemeris Message	Replace MSG-ID with UNIQUE-ID, remove LENGTH descriptor for DATA	2016	5.5.1.2 Attitude Ephemeris Message (AEM) Contents	281
Orbit Parameter Message	Replace MSG-ID with UNIQUE-ID, remove LENGTH descriptor for DATA	2016	5.5.2.1 Orbit Parameter Message (OPM) Contents	283
Orbit Mean-Elements Message	Replace MSG-ID with UNIQUE-ID, remove LENGTH descriptor for DATA	2016	5.5.2.2 Orbit Mean-Elements Message (OMM) Contents	285
Orbit Ephemeris Message	Replace MSG-ID with UNIQUE-ID, remove LENGTH descriptor for DATA	2016	5.5.2.3 Orbit Ephemeris Message (OEM) Contents	287
Tracking Data Message	Replace MSG-ID with UNIQUE-ID, remove LENGTH descriptor for DATA	2016	5.5.3 Tracking Data Messages (TDM)	289

For a short descriptive summary of the messages, see Section 4.5.1 Message Categories and Message Overview.

5.2 Control and Monitor Level Messages

5.2.1 General Information and Notification Messages

5.2.1.1 Log Message

A Log Message is time-tagged text generated by an application to notify the operator that a ground system or satellite event has occurred. Log Messages can be as trivial as those giving user confirmations but also used to convey the severity of a situation. An example of a trivial type of Log Message is one that notifies the user that a display page request is “complete”. Log Messages can also provide error information such as device failures or operator input errors. Another type of Log Message is one that identifies a certain occurrence or event has happened, for example, Loss-of-Signal is detected for a satellite data stream. If Log Messages are saved in an archive, they can provide a wealth of data as well as a chronological history of the ground system activities. This audit trail can be very useful in troubleshooting and reporting.

Table 5-5. Log Message Summary

Sender	Any GMSEC compliant application
Senders Intended Usage	Publish
Receiver	Expert Subclass, Alert Subclass, and Assessment Subclass that uses Log Messages
Receivers Intended Usage	Subscribe
What	Log action or event for display/archive/data mining/reporting/etc.
When	As needed
Quality of Service	Reliable

Example:

1. Any component needing to disseminate information should publish a Log Message
2. A Message Logger application may subscribe to all messages to place in an archive
3. A flight dynamics application may subscribe to Flight Dynamic Notification event type
4. A display application may subscribe to Critical Severity messages

5.2.1.1.1 Log Message Information Bus Header

The abbreviated table below shows the required Values of the MESSAGE-TYPE and MESSAGE-SUBTYPE fields for this message.

Table 5-6. Log Message Information Bus Header

Field Name	Req/ Opt/API	Value	Type	Notes
Message Information				
HEADER-VERSION	R	2010	F32	Version Number for this message description.
MESSAGE-TYPE	R	MSG	Header string	Message type identifier: REQ, RESP, or MSG
MESSAGE-SUBTYPE	R	LOG	Header string	Unique message identifier, fixed for GMSEC Standard Messages
More ... Please refer to Table 4-9. GMSEC Information Bus Header for a complete definition.

5.2.1.1.2 Log Message Contents

Table 5-7. Log Message Contents

Field Name	Req/ Opt	Value	Type	Notes
STRUCTURE: Body Content Identification				
CONTENT-VERSION	R	2016	F32	Version Number for this message
UNIQUE-ID	A		Header String	Unique ID used to distinguish the message
Content Body Message Specific Segment				
SUBCLASS	R	See Table 4-28. Software Class and Subclass Categories	Header string	Subclass generating the log message (or applicable subsystem of which the log message belongs)
OCCURRENCE-TYPE	R	See Tables that Follow	Header string	An occurrence types that categorizes the kind of activity or event that happened, triggering the log message
SEVERITY	R	Value	I16	Indicates the severity of the Log Message. Scale traditionally applied to message based on requirements and characteristics of the component or ground system. The severity may be used to alert the operator in some way such as visual or audible notification. Debug is typically used by software developers
		0		
		1		
		2		
		3		
		4		
USER	O		Header string	Which user/workposition/proc the message has to do with
SPACECRAFT-TIME	O		Time	Time event happened (may be earlier than actual posted time)
EVENT-TIME	R		Time	Time event happened (may be earlier than published time)
REFERENCE-ID	O		Header string	A local index or map to a table (or database) of additional information
MSG-TEXT	R		String	Text for display (typically about 60 characters)
MSG-TEXT-DETAILS	O		String	One or more paragraphs that includes more detail. Suggested corrective action. Suggest specifying url in this field
SPECIAL-INFO	O		Binary (Blob)	TBD, application use

NOTE: The REFERENCE-ID field should uniquely identify the Log Message within the defined context of the mission. That is, all Log Messages within an enterprise or system should be identified and contain an index or unique ID that can be used to reference additional information about the message (such as a full description, recommended action, or other useful information). This field can optionally be used as element *ME6* in the message subject for a very specific log message filter.

The following tables list suggested Log Message occurrence types. The OCCURRENCE-TYPE field is a required field. The tables contain lists of ground system occurrences that could be used to identify the event that triggered the generation of the Log Message. The component is not required to put out a Log Message for each type of occurrence or state change. But, if the Log Message is issued, the suggested format is strongly recommended. By implementing common Log Message occurrence types along with the recommended formats, the following goals can be achieved for Log Messages, developers, and mission operations for their deployed systems.

- **Decipherable** – Notifications through Log Messages should be readable, self-explanatory, and easily recognizable whether in a display or hardcopy report.
- **Discernable** – Log Messages should be easy to parse or extract information from, particularly the MSG-TEXT field, or any field for that matter (such as the OCCURRENCE-TYPE). Putting form to the MSG-TEXT field for common or important Log Messages will make them understandable, the situation more apparent, and consequently, easier to act upon.
- **Deterministic** – Once a Log Message has been recognized and information extracted from it (and others to provide context), decision-making and taking action becomes more perceptible. It also becomes easier to pre-define the actions and program them into clear, unambiguous rules.

A final benefit to standard looking Log Messages is that the combined GMSEC compliant components provide more uniform look-and-feel to the users making it easier for multi-system training across mission operation centers and even enterprise centers.

This is not a complete list of occurrence types. Additional values can be added as necessary.

5.2.1.1.2.1 Pass Related Occurrence Types

Table 5-8. Pass-Related Occurrence Types

Value	Occurrence Type	Occurrence Description
PREPASS	Pre-Pass	Period of time prior to start of a pass. Allows for allocation, setup, and check out of resources, communication pathways, and general preparation. Could be ~5-10 minutes in length.
PASSSTART	Planned Start Time of Pass	Time the pass is scheduled to start. (Scheduled AOS time).
AOS	Acquisition of Signal	Actual time of the AOS when ground (antenna) receives the signal. (Could be the time in mission operation center when first data is received.)
LOS	Loss of Signal	Actual time the data drops out or ceases transmission.
PASSEND	Planned End Time of Pass	Time the pass is scheduled to end. (Scheduled LOS time).
POSTPASS	Post-Pass	Period of time immediately after the LOS to wrap up the pass activities. Could include deallocation of resources, producing pass summary reports, initiation of offline data processing, and so on. Could last a few minutes or until next Pre-Pass.

For pass-related occurrences, it is recommended that the MSG-TEXT portion of the Log Message contain the Occurrence Type value and the EVENT-TIME value in the format of:

[OCCURRENCE-TYPE] Time: [EVENT-TIME]

Exhibited in the following examples:

PREPASS Time: 2014-74-16:18:15
 PASSSTART Time: 2014-74-16:28:15
 AOS Time: 2014-74-16:28:16
 PASSEND Time: 2014-74-16:46:30
 LOS Time: 2014-74-16:46:40
 POSTPASS Time: 2014-74-16:47:20

Sample Log Message Display Page

Publish Time	Msg Type	Msg Subtype	Component	Msg-Text
2014-74-16:28:17	Msg	Log	FEP	AOS Time: 2014-74-16:28:16
2014-74-16:28:33	Msg	Log	FEP	LOS Time: 2014-74-16:51:32

5.2.1.1.2.2 Telemetry Limit Violation Occurrence Types

Table 5-9. Telemetry Limit Violation Occurrence Types

Value	Occurrence Type	Occurrence Description
RED	Red Limit	Reports mnemonic entering or exiting red limit condition
YEL	Yellow Limit	Reports mnemonic entering or exiting yellow limit condition
NORM	Normal Range	Reports mnemonic returning to a normal, or within range, condition.

For telemetry limit violation notices, it is recommended that the MSG-TEXT portion of the Log message contain the following information.

[term for value] [tlm word #] [mnemonic] [violation description] “the” [limit description] “Limit of” [threshold value] “with a value of” [mnemonic value] [mnemonic units]

Exhibited in the following examples:

LRV #102 BATVOLT1 exceeded the Lower Yellow Limit of –12 with a value of –13 volts

TLM #156 BATVOLT2 is above the Upper Red limit of 15 with a value of 15.75 volts

CVT #156 BATVOLT3 is within the normal limit of 16 with a value of 14.75 volts

Note:

LRV - Last Received (or recorded) Value

TLM – Telemetry

CVT – Current Value Table

5.2.1.1.2.3 Command Verification Occurrence Types

Table 5-10. Command Verification Occurrence Types

Value	Occurrence Type	Occurrence Description
XFRD	transferredToRange	The network that connects the ground system to the spacecraft has received the command. (Comes from something other than the spacecraft.)
SENT	sentFromRange	The command has been transmitted to the spacecraft by the network that connects the ground system to the spacecraft. (Verifier comes from something other than the spacecraft.)
RCVD	Received	The SpaceSystem has received the command.
ACPT	Accepted	The SpaceSystem has accepted the command.
QUED	Queued	The SpaceSystem has scheduled the command for execution.
EXEC	Executing	The command is being executed.
COMP	Complete	Command is considered complete.
FAIL	Failed	The command failed.
TIMEOUT	Timeout	Time expired for the command to complete. A specific instance of failed.

The values in the table above were taken from the master schema for the OMG Space Domain Task Force XML Telemetric and Command Exchange (XTCE) format. This document is found at <http://www.omg.org/space/xtce>.

No recommended format has been defined for this occurrence type.

5.2.1.1.2.4 Miscellaneous Occurrence Types

Table 5-11. Miscellaneous Occurrence Types

Value	Occurrence Type	Occurrence Description
CFG	Configuration Change	Reports that the physical or logical configuration of the equipment and/or software has changed.
DIR	Directive	The echo of a directive message issued by the operator, command procedure, command schedule, or automated process
FDN	Flight Dynamics Notification	Reports completion of flight dynamics process or product
ORB	Orbital Event	Reports calculated orbital event such as orbit number, ascending/descending node, eclipse state
PROD	Product Available and/or generated	Reports that a product has been generated and is available to access
SAT	Satellite	Indicates/reports activity occurred on the satellite
SYS	System/Software	Reports detected system/software error or unexpected condition

DIR

When a Log Message is used to echo a Directive Request Message, the following is recommended.

	Retrieve From Here	And Insert Into Here
Message	Directive Request Message	Log Message
Field	DIRECTIVE-STRING	MSG-TEXT

PROD

When a Log Message is used to echo a Product Message, the following is recommended.

	Retrieve From Here	And Insert Into Here
Message	Product Message	Log Message
Field	TIME-COMPLETED	EVENT-TIME
Field	PROD-NAME PROD-TYPE PROD-SUBTYPE PROD-DIST-METHOD URI	MSG-TEXT

The MSG-TEXT format of the Log Message would be in the following format:

“Product Type/Subtype:” [PROD-TYPE] / [[PROD-SUBTYPE], “Created:”
[TIME-COMPLETED], “Available By:” [PROD-DIST-METHOD] {- URI}

Exhibited in the following examples:

Product Type/Subtype: PAS / Contact Schedule, Prod ID:
WhiteSands124, Created: 2014-123-14:32:25, Available By: URI -
Facility.Node.Computer.Disk.directory.filename

Product Type/Subtype: PAS / Schedule, Prod ID: SCHED124, Created:
2014-123-14:32:25, Available By: PROD REQ

5.2.1.1.3 Log Message Subjects

Table 5-12. Log Message Subject Naming

	Subject Standard	Mission Elements		Message Elements		Miscellaneous Elements					
Subject Element	Specification	MISSION	SAT	TYP	SUB TYP	ME1	ME2	ME3	ME4	ME5	ME6
Subject Content	GMSEC	[mission]	[sat]	MSG	LOG	[Component name of publisher]	[Subclass: ARC, CFG, CMD, DIR, ... TLM]	[Occurrence: AOS, LOS, ...RED, ...YEL]	[Severity: 1-routine, 2-med, ..., 4-crit.]	[user, Optional]	[ref ID, optional]
Example for Publisher / Sender	GMSEC	MSSN	SAT1	MSG	LOG	TLM3	TLM	RED	4	ws3	794
Example for Publisher / Sender	GMSEC	MSSN	SAT1	MSG	LOG	TLM3	CMD	CMDV	4	ws5	123
Example for Subscriber / Receiver	GMSEC	MSSN	*	MSG	LOG	*	*	*	*	>	

Table 5-13. Properties of the *Miscellaneous Elements* for the Log Message

Miscellaneous Element	Required / Optional	Description	Field in Msg, if applicable
ME1	Required	Component name of publisher	"COMPONENT" from Bus Header of msg
ME2	Required	The subclass of the log message	"SUBCLASS" from msg content
ME3	Required	The occurrence type of the event	"OCCURRENCE-TYPE" from msg content
ME4	Required	The severity of the log message	"SEVERITY" from msg content
ME5	Optional	The user or workposition originating the log message	"USER" from msg content
ME6	Optional	A reference ID assigned to the log message	"REFERENCE-ID" from msg content

Examples for Publisher / Sender:

App1, TLM2, and TLM3 each send a Log Message.

GMSEC.MSSN.SAT1.MSG.LOG.APP1.TLM.RED.4
GMSEC.MSSN.SAT1.MSG.LOG.APP1.CMD.CMDV.4.WS3.794
GMSEC.MSSN.SAT1.MSG.LOG.TLM2.TLM.RED.4.DECOM1.456
GMSEC.MSSN.SAT1.MSG.LOG.TLM3.SCH.1.ROUTINE

Examples for Subscriber / Receiver:

GMSEC.*.MSG.LOG. >
GMSEC.MSSN.*.MSG.LOG. >
GMSEC.*.SAT1.MSG.LOG.TLM2. >
GMSEC.MSSN.*.MSG.LOG.*.*.4. >

Note that since some elements are optional, it is best to subscribe with the ">" character to ensure capturing all messages.

5.2.1.2 Archive Message Retrieval

Archive Message Retrieval messages provide access to messages, excluding telemetry data, stored in a GMSEC message archive. The Archive Message Retrieval Request is used to request messages from the message archive by either

1. Providing a specific query-like statement to be used directly by the responder to access the underlying data storage mechanism. This could be
 - a. An SQL statement for a database
 - b. A grep statement to search for strings inside a file
 - c. A Perl script statement, and so on.
2. Providing a time range and pairs of message types/subtypes for a coarse description and gathering of messages.

The Archive Message Retrieval Response returns the status of the Request and the location of the output product. The output product (one file) is not returned within the response message. (Though telemetry messages could be archived and retrieved, they are categorized and treated separately since their data requires specialized processing.)

Please see Section 4.2 GMSEC Messages: Their Characteristics and Interactions for a general discussion on these types of messages.

A central or singular GMSEC message archive has not been defined for the GMSEC architecture. Any number of GMSEC messages may be archived by any number of components. For example, one component may only archive GSMEC Log messages. Another component may archive Log messages and all Directive Request messages. Also, components that archive messages do not necessarily have to provide a service to other components to extract those messages. The messages may be for that component's internal use only. How the GMSEC messages are archived and organized is left up to the mission. Lastly, as inferred from above, the method of storing the messages is not declared. A database, flat text files, or any means could be used.

5.2.1.2.1 Archive Message Retrieval Request

The Archive Message Retrieval Request is a service request issued when an application desires messages from a GMSEC message archive. The component issues an Archive Message Retrieval Request to an Archive provider component that has access to a GMSEC message archive. The request specifies the time range and message types, or a storage-specific statement for pattern matching and extraction.

Table 5-14. Archive Message Retrieval Request Summary

Sender	Any GMSEC compliant application
Senders Intended Usage	Request
Receiver	Any GMSEC compliant application that provides access to a message archive, i.e. GREAT.
Receivers Intended Usage	Subscribe
What	GMSEC messages
When	As needed

Example:

1. A GMSEC Archive and Assessment component may request archived messages, such as the Log Message for limit violations, for the generation of a report or printout.
2. A GMSEC Planning and Scheduling component may request archived messages for the re-planning of a schedule, or may even request future event messages that have been archived.
3. An analyst may want all archived messages in a specific time window for further problem analysis.

The requestor of messages will not necessarily know the format, style, or method used to store the messages. The following sections describe the two methods available to specify parameters for pattern matching. One method or the other should be used, but not both.

Extraction by Expression

If the requestor knows how the information is stored in the responder's repository, it may use that knowledge to specify a character string query. For example, if the repository is kept in a database, an SQL query string could be provided that the responder could plug directly into the database. Likewise, if the information is stored / kept in string or character "flat files", a text matching and extraction command could be applied, such a Unix grep command.

The REQ-STRING field (shorthand method) is to be used to specify the specific command for pattern matching and extraction when the requestor has that knowledge. If knowledge of the storage method is not known, then the longhand method of specifying a time range and type/subtype pairs is to be used.

Extraction by Field

The other method for identifying, matching, and extracting messages from the archive is to specify a few key fields in the messages. This method will result in a coarse selection of messages for perusal. It is not meant to identify a specific value of a specific field of a specific message (though that is possible in some circumstances). Rather, a group of associated messages will be located, extracted, and returned.

The Archive Message Retrieval Request consists of an information bus header and the message contents. The information bus header identifies the message as a GMSEC Archive Message Retrieval Request. The message contents specify the parameters for extraction out of an archive. They are:

- Time range
 - Type of time: spacecraft or a GMSEC standard format
 - Start and stop times of messages from which to retrieve
- Messages to examine
 - Type and Subtype pairing
 - Name of field in the messages that contains the time to use for the time range
- Fields in messages to pattern match
 - Other fields in the message to extract the contents and pattern match

Other fields in the message content specify the output. They are:

- Location of the one output file for the product
 - URI location to store the output file
 - File name of output file
- File description
 - Maximum size of output file to not exceed
 - Format of the output file
 - Version number of the format

A product may normally consist of multiple files. GMSEC messages are defined to deliver one product that could consist of a number of files. In this case, only one file is expected for the output product. The requestor of archived messages has the option of getting the resulting product file in one of three ways. They are:

- by reference within the response message
- included within the response message
- both

The GMSEC standard messages that may be archived contain a varying number of time fields. Some are spacecraft time and some are in the GMSEC standard time format. Also, times in messages can be the actual occurrence time of an event, the requested execution time, or the publish time of a message. These times are naturally named differently. Therefore, the Request message must specify the name of the time field in the message to use for the boundaries of the time range. A sampling of the various time fields in the GMSEC messages is provided in the following table.

Table 5-15. Examples of Time Fields in GMSEC Messages, 1 of 2

Grouped by Message Function

	Info. Bus Header	Message Content Portion							
Messages	Publish-Time	S/C Time	Event Time*	Requested Execution Time	Requested Expiration Time	Time Completed	Start-Time	Stop-Time	MNEMONIC. n.SAMPLE.n. TIME-STAMP
Log	√	√	√						
Archive Msg. Retrieval Request	√						√	√	
Archive Msg. Retrieval Response	√					√			
Directive Request	√			√	√				
Directive Response	√					√			
C2CX Messages	√								
Telemetry Data Msg.	NA								
Replay Telemetry Request	√						√	√	
Replay Telemetry Response	√					√			
Mnemonic Value Request	√								
Mnemonic Value Response	√					√			√
Mnemonic Value Data Msg.	√								√
Archive Mnemonic Value Request	√						√	√	
Archive Mnemonic Value Response	√					√			
Archive Mnemonic Value Data Msg.	√								√
Database Attributes Request	√								
Database Attributes Response	√					√			
Product Request	√						√	√	
Product Response	√					√			
Product Msg.	√					√			
Simple Service Request	√			√	√				
Simple Service Response	√					√			

	Info. Bus Header	Message Content Portion							
Messages	Publish-Time	S/C Time	Event Time*	Requested Execution Time	Requested Expiration Time	Time Completed	Start-Time	Stop-Time	MNEMONIC. n.SAMPLE.n. TIME-STAMP
Attitude Parameter Message	√		√						
Attitude Ephemeris Message	√		√						
Orbit Parameter Message	√		√						
Orbit Mean-Elements Message	√		√						
Orbit Ephemeris Message	√		√						
Tracking Data Message	√		√				√	√	

* or Creation Time

Table 5-16. Examples of Time Fields in GMSEC Messages, 2 of 2

	Info. Bus Header	Message Content Portion				
Messages	Publish-Time	Release Time	Earliest Uplink Time	Latest Uplink Time	Spacecraft Execution Time	Time Completed
Command Request	√	√	√	√	√	
Command Response	√	√				√

5.2.1.2.1.1 Archive Message Retrieval Request Information Bus Header

The abbreviated table below shows the required Values of the MESSAGE-TYPE and MESSAGE-SUBTYPE fields for this message.

Table 5-17. Archive Message Retrieval Request Information Bus Header

Field Name	Req/ Opt/API	Value	Type	Notes
Message Information				
HEADER-VERSION	R	2010	F32	Version Number for this message description.
MESSAGE-TYPE	R	REQ	Header string	Message type identifier: REQ, RESP, or MSG
MESSAGE-SUBTYPE	R	AMSG	Header string	Unique message identifier, fixed for GMSEC Standard Messages
More ... Please refer to Table 4-9. GMSEC Information Bus Header for a complete definition.

5.2.1.2.1.2 Archive Message Retrieval Request Content

Table 5-18. Archive Message Retrieval Request Contents

Field Name	Req/Opt	Value/Description		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message
UNIQUE-ID	A			Header String	Unique ID used to distinguish the message
STRUCTURE: Time Window					
START-TIME	R			Time (absolute or relative)	Requested start time of the messages to be retrieved from the Message Archive
STOP-TIME	O			Time (absolute or relative)	Requested stop time of the messages to be retrieved from the Message Archive. Defaults to the end of the Message Archive.
STRUCTURE: Product Distribution Options					
DELIVER-VIA-REFERENCE	O	Value	Description	Boolean	Indicates if the data will be referenced by a URI in the single response message. Defaults to No.
		0	No / False		
		1	Yes / True		
DELIVER-VIA-INCLUDE	O	0	No / False	Boolean	Indicates if the data is to be included in the single response message. Defaults to Yes.
		1	Yes / True		
		STRUCTURE: Output Product Category Identification			
PROD-NAME	O			String	Name of the product being requested
PROD-DESCRIPTION	O			String	Description of the product in text or xml
PROD-TYPE	O	Value	Description	String	Product type and subtype being requested. (See Table 4-29. Product Categories)
		AAA	Archive and Assessment		
PROD-SUBTYPE	O	MSG	Message	String	
NUM-OF-PROD-SUBTYPES	O			U16	Number of further delineations / categories beyond the product subtype. Also, used as msg subject elements <i>ME5</i> , <i>ME6</i> , etc. in the Product Message.
PROD-SUBTYPE.n.NAME	O			String	First subcategory of the product subtype. (Subject elements <i>ME5</i> , <i>ME6</i> , etc. of the Product Message)
STRUCTURE: Output File Attributes					
URI	O			String	Location where the requesting component is asking for the product file(s) to be stored. Could be a web address, directory or folder specification
NAME-PATTERN	O			String	Describes the name of the output file
DESCRIPTION	O			String	Description of the file in text or xml
FORMAT	O			String	Describes the file format
VERSION	O			String	Identifies the version of the file
SIZE	O	Kilobytes		U32	Maximum size of the file acceptable to the requester. Size specified in KB.
Query Shorthand					
REQ-STRING	O			String	Specific to the responder / provider of the requested information. The string will define a database query, a script expression, Unix statement, or some other statement for extracting the information from the provider's repository.
Query Longhand					
NUM-OF-MSGS	D	Required if using longhand query.		U16	Indicates the number of different message type / subtype pairs requested from the Message Archive.

MSG.n.TYPE	D	Required if using longhand query. "n" starts at "1".	String	Message Type/Subtype pairing to identify the message to be retrieved from the archive
MSG.n.SUBTYPE	D	Required if using longhand query.	String	
MSG.n.TIME-FIELD-NAME	O		String	Name of field in the message that contains the time to examine. Will default to PUBLISH-TIME in Info. Bus Header.
MSG.n.TIME-TYPE	O	Value	Description	Indicates the format of the time to examine in the retrieved messages. Defaults to GMSEC standard time format.
		0	Spacecraft Time	
		1	GMSEC std. Time	
MSG.n.NUM-OF-FIELDS	O		U16	Number of message fields to examine and match for retrieval from the archive
MSG.n.FIELD.n.NAME	O		String	Name of the message field to match for retrieval from the archive
MSG.n.FIELD.n.CONTENT	O		String	Contents of the message field used in matching the messages for retrieval from the archive

START-RETRIEVAL-TIME and STOP-RETRIEVAL-TIME:

The requestor could specify the START and STOP times in the following ways. Either or both the start and stop times must be absolute.

Table 5-19. Examples of Start and Stop Times

Start Time	Stop Time	Example	Description
Absolute	Absolute	START: 2014-123-14:30:00 STOP: 2014-123-14:30:10	Boundaries of the Start and Stop time have been exactly specified.
Absolute	Relative (duration)	START: 2014-123-14:30:00 STOP: +10:00	Time window has an absolute start time and extends for 10 minutes.
Relative (duration)	Absolute	START: -10:00 STOP: 2014-123-14:30:00	Time window will end at a specified stop time and start 10 minutes prior to that.
Relative	Relative	Not Applicable	Unless there has been some previously established baseline time upon which to offset the duration times (such as "now"), this combination is ambiguous.

5.2.1.2.2 Archive Message Retrieval Response

An archive message provider in response to an Archive Message Retrieval Request will send an Archive Message Retrieval Response. The archive message provider must return the status of the action completed along with the location of the Archive Retrieval Message file product. A series of Archive Message Retrieval Responses may be required. In this case, an initial acknowledgement response message is issued, followed by interim or interactive “working” response type messages to let the requesting application know that the request is still being processed, and finally by a completion response type message. If an operator or audit trail notification is required, the requesting application is responsible for generating a GMSEC Log Message indicating the result of the Archive Message Retrieval Request. Please see Section 4.2 GMSEC Messages: Their Characteristics and Interactions for a general discussion on these types of messages.

Table 5-20. Archive Message Retrieval Response Summary

Sender	Application that received the Archive Message Retrieval Request
Senders Intended Usage	Publish or Reply
Receiver	Application that issued the Archive Message Retrieval Request
Receivers Intended Usage	Subscribe
What	Provide success/failure response to the service that was requested and the knowledge to access the extracted messages
When	Upon receipt of the Archive Message Retrieval Request, on an interval for those services that are time-consuming, and on completion.
Quality of Service	Reliable

Example:

1. Acknowledge receipt of an Archive Message Retrieval Request
2. Indicate the Archive Message Retrieval Request is still being processed
3. Indicate the Archive Message Retrieval Request has successfully completed

5.2.1.2.2.1 Archive Message Retrieval Response Information Bus Header

The abbreviated table below shows the required Values of the MESSAGE-TYPE and MESSAGE-SUBTYPE fields for this message.

Table 5-21. Archive Message Retrieval Response Information Bus Header

Field Name	Req/ Opt/API	Value	Type	Notes
Message Information				
HEADER-VERSION	R	2010	F32	Version Number for this message description.
MESSAGE-TYPE	R	RESP	Header string	Message type identifier: REQ, RESP, or MSG
MESSAGE-SUBTYPE	R	AMSG	Header string	Unique message identifier, fixed for GMSEC Standard Messages
More ... Please refer to Table 4-9. GMSEC Information Bus Header for a complete definition.

5.2.1.2.2.2 Archive Message Retrieval Response Contents

Table 5-22. Archive Message Retrieval Response Contents

Field Name	Req/ Opt	Value		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	
STRUCTURE: Response Status					
RESPONSE-STATUS	R	Value	Description	I16	Identifies the status of the Archive retrieval message Request being processed
		1	Acknowledgement		
		2	Working/keep alive		
		3	Successful completion		
		4	Failed completion		
		5	Invalid Request		
	6	Final Response			
TIME-COMPLETED	O			Time	Time application completed processing the request
RETURN-VALUE	O	Value	Description	I32	Return value or status based on the RESPONSE-STATUS. Used to indicate product URI as requestor or responder. Also can be used to provide function call status or error code in the case of failed completion
		1	Product file placed in URI specified by requestor		
		2	Product file placed in URI specified by responder		
		Other	Error code of a failed completion		
STRUCTURE: Output Product Category Identification					
PROD-NAME	O			String	Name of the product being returned
PROD-DESCRIPTION	O			String	Description of the product in text or xml
PROD-TYPE	O	Value	Description	String	Product type and subtype being requested. (See Table 4-29. Product Categories)
		AAA	Archive and Analysis		
PROD-SUBTYPE	O	MSG	Message	String	
NUM-OF-PROD-SUBTYPES	O	0+		U16	Number of further delineations / categories beyond the product subtype. Also, used as msg subject elements <i>ME5</i> , <i>ME6</i> , etc. in Product Message.
PROD-SUBTYPE.n.NAME	O			String	First subcategory of the product subtype. (Subject elements <i>ME5</i> , <i>ME6</i> , etc. of the Product Message)
STRUCTURE: Output File Attributes					
URI	O			String	URI specifying the location where the file product is stored
NAME-PATTERN	O			String	Describes the name of the file
DESCRIPTION	O			String	Description of the file contents in text or xml
FORMAT	O			String	Describes the file format of the file
VERSION	O			String	Indicates the version of the file
SIZE	O	KB		U16	Actual size of the file
STRUCTURE: Data File					
DATA	O			Binary (blob)	The file content

A product consisting of a single file is returned in the response message. The file contains the messages that satisfied the criteria specified in the Archive Message Retrieval Request Message.

Table 5-23. Meaning of Response Status and Return Value with Recommended Actions

User Specified the URI	RESPONSE-STATUS	RETURN-VALUE	URI	Action
N	Successful	2 (The only meaningful value)	Product was generated and placed in URI chosen by responder	Requestor should retrieve file at responder's URI location
Y	Successful	1	Product was generated and placed in URI specified by requestor	Requestor should retrieve file at the specified URI
Y	Successful	2	Product was generated but placed in alternate URI chosen by responder	Requestor should retrieve file at responder's URI location
Y or N	Failed	3	Product exceed maximum requested file size or the default maximum file size	

5.2.1.2.3 Archive Message Retrieval Subjects

Table 5-24. Archive Message Retrieval Request Subject Naming

	Subject Standard	Mission Elements		Message Elements		<i>Miscellaneous Elements</i>		
Subject Element	Specification	MISSION	SAT	TYP	SUBTYP	ME1	ME2	ME3
Subject Content	GMSEC	[mission]	[sat]	REQ	AMSG	[Component of responder: ARCHIVER, ANALYZER, ...]		
Example for Publisher / Sender	GMSEC	MSSN	SAT1	REQ	AMSG	ARCHIVER		
Example for Publisher / Sender	GMSEC	MSSN	SAT1	REQ	AMSG	ANALYZER		
Example for Subscriber / Receiver	GMSEC	*	*	REQ	AMSG	ANALYZER		

Table 5-25. Properties of the *Miscellaneous Elements* for the Archive Message Retrieval Request

<i>Miscellaneous Element</i>	Required / Optional	Description	Field in Msg, if applicable
ME1	Required	Component name of Responder	NA
ME2	Not used		
ME3	Not used		

Examples:

Two components, ANALYZER and ARCHIVER, interact with the Archive Message Retrieval Request.

ANALYZER message subject to send the Archive Message Retrieval Request to ARCHIVER:

GMSEC.FILL.FILL.REQ.AMSG.ARCHIVER

ARCHIVER message subject to receive its own Archive Message Retrieval Request:

GMSEC.*.*.REQ.AMSG.ARCHIVER

Table 5-26. Archive Message Retrieval Response Subject Naming

	Subject Standard	Mission Elements		Message Elements		<i>Miscellaneous Elements</i>		
Subject Element	Specification	MISSION	SAT	TYP	SUBTYP	ME1	ME2	ME3
Subject Content	GMSEC	[mission]	[sat]	RESP	AMSG	[Component of Requestor: APP1 , ARCVR, ...]	[Response Status: 1-acknowledgment,...4-failed]	[Response Status: 1-acknowledgment,...4-failed]
Example for Publisher / Sender	GMSEC	MSSN	SAT1	RESP	AMSG	ARCVR	1	
Example for Publisher / Sender	GMSEC	MSSN	SAT1	RESP	AMSG	ARCVR	3	
Example for Subscriber / Receiver	GMSEC	*	SAT1	RESP	AMSG	APP1	*	

Table 5-27. Properties of the *Miscellaneous Elements* for the Archive Message Retrieval Response

<i>Miscellaneous Element</i>	Required / Optional	Description	Field Origination in Msg, if applicable
ME1	Required	Component name of Requestor	Echo of "COMPONENT" in header of Request msg
ME2	Required	Status type supplied by Responder	"RESPONSE-STATUS" from content of Response message

Examples:

Two components, APP1 and ARCVR, interact with the Archive Message Retrieval Response.

ARCVR subject to send the Archive Message Retrieval Response to APP1:

GMSEC.FILL.FILL.RESP.AMSG.APP1.3

APP1 subject to receive its own Archive Message Retrieval Responses:

GMSEC.MSSN.SAT1.RESP.AMSG.APP1.* or

GMSEC.*.*.RESP.AMSG.APP1.>

5.2.2 Action and Solicitation Messages

One of the requirements imposed by GMSEC on compliant components is the ability of components to accept control messages. (Another requirement is to send status or informational messages using the GMSEC Log messages.) The requirement for a component to accept and process control messages provides for

1. External or remote control by another component. This permits a manager or central controller component to direct, coordinate, synchronize, and in general, orchestrate the operations of a system as a whole entity. Some specialized areas of control could include pre-pass, pass, and post-pass operations, startup and shutdown, failover, and various process flows.
2. A means to request a specific functional capability of that component.

There are three kinds of GMSEC control message interactions:

1. **Component-to-Component (C2CX)** - The C2CX Control messages are those that are typically used to request “under the hood” functions that, while essential for operations, are not of general interest. A C2CX Control message might be used to send computer resource usage information or send periodic heartbeat messages.
2. **Directive** - The Directive Request message is primarily used for requests that are also to be visible to the operations staff and are directly related to the overall mission of the system. For example, a Directive Request would normally be used to page a Flight Operations Team member. Directive messages will typically include a spacecraft and test operations language (STOL) text string the receiver will parse to determine what function is being requested.
3. **Simple Services** – The Simple Service Request and Response messages operate in a manner similar to the Directive messages. Where the Directive Request message will include a text string to be parsed by the receiver, the Simple Service Request message will include a list of parameters that will identify the service being requested. The Simple Service Request Message will supply the service, operation, and list of associated parameters.

In one sense, all GMSEC Request Messages are a specialized form of the Directive or Simple Service messages. That is, each Request Message requests a function, capability, or service that is available within the system. The Directive and Simple Service messages are a generalized form of making a capability request while the other Request Messages are available to make specific

capability requests. These specific Request Messages might be created or available for high frequency requested capabilities, ease of making the request (for both the requestor and provider), high visibility and tracking of requests, or other reasons. Of course, Directive Messages are also used to make requests of components that are not recognized as general system-wide services.

In order to make a request of a component the requestor must know

1. What service(s) is available,
2. The syntax for requesting that service

On the opposite end, the provider of the service must make this information known to all potential requestors. Thus, each component, in order to accept a Directive or Simple Service Request must

1. Define those capabilities it will provide or accept requests for,
2. Define the syntax to request those capabilities,
3. Be able to parse, extract, or identify the requested capability from the request message if there has not been a specific message created to do so.

In a component's documentation that describes the capabilities, services, and products it provides, it must also describe the means and syntax to request those items. For example, if a GMSEC message exists to request that service, then the description should include

- Name of service(s) or product(s) that are available
- GMSEC messages used to request that item

For the C2CX Control message, the request, capability or function is provided in the CNTL-KEYWORD and CNTL-STRING fields. For Directive Request messages, the capability or requested service is identified in the DIRECTIVE-STRING and the DIRECTIVE-KEYWORD fields. For the Simple Service Request Message, a number of fields are provided to identify the service, operation, and any associated parameters. For other products and services, the provider must make clear to the consumers:

- What messages are to be used
- What fields are required in the messages
- What values are acceptable for the required fields

The C2CX Control and Directive messages are discussed in the sections that immediately follow. For the Simple Service, please refer to Section 5.4.2 Simple Service Messages for additional details.

5.2.2.1 Directive Messages

Directives, instructions directing a component to action, may originate from a GUI, command line, schedule, procedure, or virtually anywhere within a space-ground system. The Directive Request Message is the mechanism to send a directive to a component. The Directive Response Message is used to return an acknowledgement and status of the directive action to the originator. Please see Section 4.2 GMSEC Messages: Their Characteristics and Interactions for a general discussion on these types of messages.

Currently, it is not expected that GMSEC directives will be sent between the spacecraft and the ground, though this is not precluded. As the architecture expands and services are created for general mission operations and control, space-to-ground transactions may utilize GMSEC defined messages.

5.2.2.1.1 Directive Request Message

A Directive Request Message is the means for one application to request a service from another application. Directives themselves can be input from a user through a GUI or command line, or as part of the internal logic of a component. They can also be grouped together with logic in a file as a procedure (AKA proc). Directives can also be found in a command schedule, organized by time-tag for automatic execution. As automation increases, more and more Directive Request Messages are generated internally as certain data conditions are detected.

Table 5-28. Directive Request Message Summary

Sender	Any GMSEC compliant application
Senders Intended Usage	Request a service or function
Receiver	Application providing a service, or an application collecting directives for audit trail purposes
Receivers Intended Usage	Subscribe
What	Action or service request initiated by user, software, procedure, command schedule, etc.
When	Upon detection of data condition, upon scheduled time, upon entry
Quality of Service	Guaranteed

Examples:

1. An operator input issuing a satellite command
2. An operator input requesting a display page
3. A request to start/pause/stop/resume a schedule
4. A request to initiate pre-pass setup
5. Any request issued from a procedure

5.2.2.1.1.1 Directive Request Message Information Bus Header

The abbreviated table below shows the required Values of the MESSAGE-TYPE and MESSAGE-SUBTYPE fields for this message.

Table 5-29. Directive Request Message Information Bus Header

Field Name	Req/ Opt/API	Value	Type	Notes
Message Information				
HEADER-VERSION	R	2010	F32	Version Number for this message description.
MESSAGE-TYPE	R	REQ	Header string	Message type identifier: REQ, RESP, or MSG
MESSAGE-SUBTYPE	R	DIR	Header string	Unique message identifier, fixed for GMSEC Standard Messages
More ... Please refer to Table 4-9. GMSEC Information Bus Header for a complete definition.

5.2.2.1.1.2 Directive Request Message Contents

Table 5-30. Directive Request Message Contents

Field Name	Req/ Opt	Value	Type	Notes	
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016	F32	Version Number for this message content description	
UNIQUE-ID	A		Header String		
Source Information					
USER	O		String	Which user/workposition/proc/schedule the message is coming from	
STRUCTURE: Directive Information					
DIRECTIVE-KEYWORD	O	Uppercase	Header string	Keyword extracted from the directive string. Useful for routing/processing	
DIRECTIVE-STRING	R		String	Full directive string that includes the keyword	
SPECIAL-INFO	O		Binary	For application use	
Directive Parameters					
PRIORITY	O	Value	Description	I16	Indicates processing priority, if applicable
		1	Nominal		
		2	Medium		
		3	High		
RESPONSE	R	Value	Description	Boolean	Indicates if a response is required. Defaults to Yes.
		0	False or no response		
		1	True or must respond		
REQUESTED-EXECUTION-TIME	O		Time		Absolute or relative time can apply.
REQUESTED-EXPIRATION-TIME	O		Time		Absolute or relative time can apply.

For an explanation on how the REQUESTED-EXECUTION-TIME and REQUESTED-EXPIRATION-TIME could operate, see Table 5-19. Examples of Start and Stop Time.

5.2.2.1.2 Directive Response Message

A Directive Response Message is sent by an application in response to a Directive Request Message. The Directive Response Message will provide acknowledgment of the Directive Request Message and a status of the action completed. A series of Directive Response Messages may be required in the case where the processing of the action is lengthy. An example of this would be an archive retrieval, plot, or orbit determination calculation. In this event, an interim or interactive “working” type message would be issued to let the original application know that the action is still being processed. Please see Section 4.2 GMSEC Messages: Their Characteristics and Interactions for a general discussion on these types of messages.

Table 5-31. Directive Response Message Summary

Sender	Application that received the Directive Request Message corresponding to the Directive Response Message
Senders Intended Usage	Reply
Receiver	Application that issued the Directive Request Message or an application collecting Directive Response Messages for audit trail purposes
Receivers Intended Usage	Subscribe
What	Provide success/failure response to the service that was requested
When	Upon receipt of Directive Request Message or at intervals for those services that are time-consuming
Quality of Service	Guaranteed

Example:

1. Acknowledge receipt of a directive
2. Indicate the directive is still being processed
3. Indicate the directive has successfully completed

5.2.2.1.2.1 Directive Response Message Information Bus Header

The abbreviated table below shows the required Values of the MESSAGE-TYPE and MESSAGE-SUBTYPE fields for this message.

Table 5-32. Directive Response Information Bus Header

Field Name	Req/ Opt/API	Value	Type	Notes
Message Information				
HEADER-VERSION	R	2010	F32	Version Number for this message description.
MESSAGE-TYPE	R	RESP	Header string	Message type identifier: REQ, RESP, or MSG
MESSAGE-SUBTYPE	R	DIR	Header string	Unique message identifier, fixed for GMSEC Standard Messages
More ... Please refer to Table 4-9. GMSEC Information Bus Header for a complete definition.

5.2.2.1.2.2 Directive Response Message Contents

Table 5-33. Directive Response Message Contents

Field Name	Req/ Opt	Value	Type	Notes
STRUCTURE: Body Content Identification				
CONTENT-VERSION	R	2016	F32	Version Number for this message content description
UNIQUE-ID	A		Header String	
STRUCTURE: Response Status				
RESPONSE-STATUS	R	Value	Description	Identifies the status of the Directive being processed
		1	Acknowledgement	
		2	Working/keep alive	
		3	Successful completion	
		4	Failed completion	
		5	Invalid Request	
		6	Final Message	
TIME-COMPLETED	O		Time	Time application completed processing the directive
RETURN-VALUE	O		I32	Return value or status based on the RESPONSE-STATUS. Useful to provide function call status or error code in the case of failed completion
STRUCTURE: Data Abstract				
DATA	O		Dependent upon response	Additional data that may be desired along with the completion status

5.2.2.1.3 Directive Message Subjects

Table 5-34. Directive Request Message Subject Naming

	Subject Standard	Mission Elements		Message Elements		Miscellaneous Elements		
Subject Element	Specification	MISSION	SAT	TYP	SUB TYP	ME1	ME2	ME3
Subject Content	GMSEC	[mission]	[sat]	REQ	DIR	[Component: APP1, TLM2, TLM3 ...]	[DIRECTIVE-KEYWORD]	
Example for Publisher / Sender	GMSEC	MSSN	SAT1	REQ	DIR	APP1	[DO_ABC]	
Example for Publisher / Sender	GMSEC	MSSN	SAT1	REQ	DIR	APP2	[DO-XYZ]	
Example for Subscriber / Receiver	GMSEC	MSSN	SAT1	REQ	*	APP4	[DO_ABC]	

Table 5-35. Properties of the *Miscellaneous Elements* for the Directive Request Message

Miscellaneous Element	Required / Optional	Description	Field in Msg, if applicable
ME1	Required	Component name of Responder	NA
ME2	O	A keyword the receiving process could use for filtering	"DIRECTIVE-KEYWORD" in content of Request msg
ME3 ...	Not used		

Examples:

Two components, APP1 and APP4, interact with the Directive Request Message.

APP1 subject to send the Directive Request to APP4:

GMSEC.MSSN.SAT1.REQ.DIR.APP4

APP4 subject to receive its own Directive Request Messages:

GMSEC.MSSN.SAT1.REQ.DIR.APP4.>

APP4 subject to receive any APP4 Request Message:

GMSEC.*.*.REQ.*.APP4.>

APP4 subject to receive a specific Directive Message:

GMSEC.*.*.REQ.DIR.APP1.DO_ABC

Table 5-36. Directive Response Message Subject Naming

	Subject Standard	Mission Elements		Message Elements		Miscellaneous Elements		
Subject Element	Specification	MISSION	SAT	TYP	SUBTYP	ME1	ME2	ME3
Subject Content	GMSEC	[mission]	[sat]	RESP	DIR	[Component: APP1, TLM2...]	[Status]	[]
Example for Publisher / Sender	GMSEC	MSSN	SAT1	RESP	DIR	APP1	1	
Example for Publisher / Sender	GMSEC	MSSN	SAT1	RESP	DIR	APP4	4	
Example for Subscriber / Receiver	GMSEC	MSSN	SAT1	RESP	DIR	APP4	*	

Table 5-37. Properties of the *Miscellaneous Elements* for the Directive Response Message

Miscellaneous Element	Required / Optional	Description	Field Origination in Msg, if applicable
ME1	Required	Component name of Requestor	Echo of "COMPONENT" in header of Request msg
ME2	Required	Status type supplied by Responder	"RESPONSE-STATUS" from content of Response message

Examples:

Two components, APP4 and APP1, interact with the Directive Response message.

APP1 subject to send the Directive Response to APP4:

GMSEC.MSSN.SAT1.RESP.DIR.APP4.3

APP4 subscribes to receive its own Directive Response Messages:

GMSEC.MSSN.SAT1.RESP.DIR.APP4.* or

GMSEC.*.RESP.DIR.APP4.>

5.2.2.2 Component-To-Component Transfer (C2CX) Messages

The Component-To-Component Transfer Message provides the messaging framework to transfer control and status information between GMSEC components. The messages could be used for sending status, control, setup, initialization, heartbeat, security, a handshake, and etc., from one component to another or to multiple components (one-to-many). A few types of C2CX messages have been defined but additional types are easily added.

The C2CX messages are generally used for passing control or status information between components. The messages are also typically unidirectional. If components need back and forth interaction or confirmation, it is recommended that the Directive Request and Response messages be used. Also, if it is desirable to know the progress or status of a C2CX message, the receiver of the message can issue a Log message at noteworthy points within its processing cycle. As such, the C2CX messages act at a layer below general system knowledge that is accomplished through other GMSEC messages such as the Log message and Directive messages. The C2CX messages seek to provide a communication mechanism just under the radar of the general system, but also easily viewable, as necessary.

Table 5-38. Component-To-Component Transfer Message Summary

Sender	Any GMSEC compliant application
Senders Intended Usage	Publish
Receiver	Any GMSEC compliant application
Receivers Intended Usage	Subscribe
What	Status and Control type information
When	As needed and depends upon the type of information being transferred
Quality of Service	Reliable

Examples:

1. **C2CX CFG** (Configuration): This message is used to report software configuration information from a logical or relational perspective.
2. **C2CX HB** (Heartbeat): This message is published by all GMSEC active components and subscribed to by a monitoring component. The C2CX Heartbeat message is used to monitor the ongoing presence of the components and detect their absence.
3. **C2CX DEV** (Device): This message is used to report on the status of one or more devices, physical or virtual, or any collection of data.
4. **C2CX CNTL** (Control): A component can request another component to action with the CNTL message.
5. **C2CX RSRC** (Resource): This message is used to report a snapshot of computer performance data (CPU, memory, disk, and network usage).

5.2.2.2.1 Component-To-Component Transfer Message Information Bus Header

The abbreviated table below shows the required Values of the MESSAGE-TYPE and MESSAGE-SUBTYPE fields for this message.

Table 5-39. Component-To-Component Transfer Message Information Bus Header

Field Name	Req/ Opt/API	Value	Type	Notes
Message Information				
HEADER-VERSION	R	2010	F32	Version Number for this message description.
MESSAGE-TYPE	R	MSG	Header string	Message type identifier: REQ, RESP, or MSG
MESSAGE-SUBTYPE	R	C2CX	Header string	Unique message identifier, fixed for GMSEC Standard Messages
More ... Please refer to Table 4-9. GMSEC Information Bus Header for a complete definition.

5.2.2.2.2 Component-To-Component Transfer Message Content

The Component-To-Component Transfer Message consists of an information bus header and the message content portions. Currently, there are five subtypes of C2CX messages. They are:

- Configuration Status
- Control
- Device
- Heartbeat
- Resource

The information bus header and message content identify the C2CX message types as follows:

Table 5-40. Component-To-Component Transfer Message Subtypes

Message Portion	Info. Bus Header	Info. Bus Header	Message Content
Field	MESSAGE-TYPE	MESSAGE-SUBTYPE	C2CX-SUBTYPE
Value	MSG	C2CX	CFG
			CNTL
			DEV
			HB
			RSRC

Additional C2CX messages can be added as desired. Some examples are:
 HS – Handshake (to establish communication protocol)
 SECR – Security (pass, set, or control information)
 SET – Set a parameter or variable to a specific value (similar to CNTL)
 GROUP – Create, join, leave, and disband a dynamic group association

5.2.2.2.1 Component-To-Component Transfer Configuration Status Message

Table 5-41. Component-To-Component Transfer Message Contents for Configuration Status

Field Name	Req/Opt	Value/Description		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	Unique ID used to distinguish the message
Content Body Segment					
Body Content Subtype					
C2CX-SUBTYPE	R	Value	Description	Header String	Identifies the type of information being transferred between the Components
		CFG	Configuration Status		
MY-ROLE	R			String	Role the reporting component has in the configuration. E.g. PRIMARY, BACKUP, AGENT, SERVER, MEMBER, MGR, ...
NUM-OF-ASSOCS	O	"n" starts at "1"		U16	The number of associations to be reported.
ASSOC.n.GROUP	O			String	Name of component or group associated with
ASSOC.n.NODE	O			String	Location of associated component or group
ASSOC.n.ROLE	O			String	Role the associated component has, if known

The **CFG – Configuration Status** C2CX message is used by software components to report their configuration information. (The **DEV – Device** C2CX message is used to report the status of devices.) Note that the Information Bus Header already contains information about the component, namely:

- Support of mission and satellites
- Name and location (facility and node)
- Class of capabilities provided

The additional information to be passed or reported is non-standard. The components reporting the configuration information and the components monitoring the configuration will need to establish

- What configuration information is to be reported (and name the fields)
- When it is to be reported (upon change or periodically?)
- What format to report the configuration information

A monitoring agent would collect Heartbeat messages and Configuration Status messages and possibly the Device messages to maintain a picture of what

software is operating where, in what capacity, in what state, and with what physical and/or logical associations. The collected information can be

- Presented in a graphical colored display depicting the operating environment and the logical associations of the components
- Used to determine what pre-determined actions to take in the event of a failure or degraded operations

As an example, a front-end telemetry and command processor would publish a C2CX CFG message whenever it is first run and thereafter when it associates (or disassociates) itself with another cooperating component. In addition to the information already provided in the Information Bus Header, it would also report the following configuration information:

- The role of the reporting component (PRIMARY, BACKUP)
- Number of associations
 - Name of component or port associated with, such as
 - Telemetry and command processor (decommutation and command verification)
 - External telemetry and command link / port
 - Planner and Scheduler (to direct the setup and operation of a pass)
 - Flight dynamics component (to exchange downlink or other information)
 - Node of the associated component, if known
 - Role of the associated component (PRIMARY, BACKUP)

If the monitoring agent is also a configuration manager, it can establish the present operating configuration and also prepare a contingent configuration. In the event of a component failure or processor failure, the configuration manager will know if it has the required and sufficient number of components to sustain operations. If not, it can also determine if it has the required and necessary numbers of components should a failover or restart procedure be invoked, and if so, automatically initiate that procedure.

The minimum and sometimes maximum configuration information a component can report is its own role. A single component with no associations would normally report its role as PRIMARY. Some configuration information may already be known and available if the components used a pre-registration or registration mechanism to disclose such information as:

- Nodes where they can execute
- Are they standalone or redundant, and if so, on what nodes could the redundant component operate

Furthermore, the Configuration Status Message may be used to report group associations. That is, to what group does this component belong, and is it a member and/or a manager of the group. Some groups of components may operate in a peer only association where other groups may require a group manager for organization, control, and direction.

Groups can be used for a number of purposes. These include, but are not limited to:

- **Message Exchange**
Groups can be formed to pass messages in a number of relationships and locations that include:
 - Peer-to-Peer
 - Client-Server
 - Manager-Member
 - Local and Distributed
- **Configuration Management**
Equipment can be logically associated to form groups (or suites or strings) that must operate together, failover together, have a minimum configuration (quorum), addressed as a group, or other operating constraints or configurations.

Group formation can be

- Pre-defined
- Dynamic
If dynamic, then additional group functions may be required, such as:
 - Create / Disband
 - Join / Leave

Groups can also be hierarchical as in the following table:

Table 5-42. Group Hierarchical Associations

Grouping Level	Space	Ground
Software	Software Application	Software Application
Hardware / Equipment	Processor	Computer
	Bus	Bus/LAN/WAN/Web
	Satellite	Facility/Center
	Constellation	Enterprise
Business	Mission	

The above discussion and illustrations provide a sampling of the ways groups may be employed within the messaging framework.

5.2.2.2.2 Component-To-Component Transfer Control Message

Table 5-43. Component-To-Component Transfer Message Contents for Control

Field Name	Req/Opt	Value/Description		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	Unique ID used to distinguish the message
Content Body Segment					
Body Content Subtype					
C2CX-SUBTYPE	R	Value	Description	Header String	Identifies the type of information being transferred between the Components
		CNTL	Control		
STRUCTURE: Directive Information					
CNTL-KEYWORD	O	Uppercase		Header String	Keyword extracted from the CNTL-STRING. Useful for routing/processing.
CNTL-STRING	O			String	Parameters to guide the component on further processing. E.g., INIT, Stop, Shutdown, Restart, Do X, Y, and Z.
SPECIAL-INFO	O			Binary	For application use. Any additional information can be provided here.

The **Control** C2CX message is used to start, restart, reinitialize, or otherwise control another component. As an example, components may determine that they will not proceed with their processing until they have received a C2CX Control message with a CNTL-STRING of "INIT" or "START". Other components may require additional information in CNTL-STRING to begin processing. Still other components that have been performing their processing may allow themselves to be re-directed in their processing. Upon the reception of a Control C2CX message, a component will re-direct itself according to the supplied string of parameters.

For instance, a component could request another component to change its rate of publishing a Heartbeat message. The requesting component would send a C2CX message with a C2CX-SUBTYPE of CNTL and with a CNTL-STRING of "SET HB 15" (or some known decipherable command string).

Missions may want to have different processing modes or signals when the course of events changes what standard actions are to follow. For example, a system wide indicator may be sent using the C2CX CNTL messages with a value for the CNTL-STRING to signify that a pass has begun and the processing mode is 'PASS'. Or, the processing mode is 'PRE-PASS', 'POST-PASS', 'LIGHTS-OUT', 'LIGHTS-ON', 'AUTONOMOUS', 'SIMULATION', 'LAUNCH', 'ECLIPSE-PERIOD', 'MANEUVER', 'SAFE-MODE', or any such state that could affect some components and result in conditional processing or decision making.

In these examples, a monitoring agent, criteria action agent, decision making component, script control, or processing manager would monitor the events for

state changes and then issue the C2CX CNTL message for all or a subset of the components.

A further example could involve distributed simulations. A key factor in these simulations is to know the simulated time. A C2CX CNTL message can be defined to set, distribute, or synchronize components to a simulated time. The CNTL message might be used to set the time or advance the simulated time by a delta time. This includes training, development, integration, and pre-launch / operations simulations. If necessary, a separate C2CX message may be developed with a C2CX-SUBTYPE of SETTIME with associated parameters.

Finally, a simple application could be a PING function. "PING" placed in the CNTL-STRING field would simply require the receiver to publish the same type of message but with "PING-ACK" in the CNTL-STRING field. Alternately, separate C2CX subtypes could be defined, the PING and PING-ACK subtypes.

5.2.2.2.3 Component-To-Component Transfer Device Message

Table 5-44. Component-To-Component Transfer Message Contents for Device

Field Name	Req/Opt	Value/Description		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	Unique ID used to distinguish the message
Content Body Segment					
Body Content Subtype					
C2CX-SUBTYPE	R	Value	Description	Header String	Identifies the type of information being transferred between the Components
		DEV	Device		
Device Status Information					
NUM-OF-DEVICES	R	1+		U16	Number of devices being reported in this message
DEVICE.n.NAME	R	“n” starts at 1		String	Name of the device
DEVICE.n.NUMBER	O			I16	A number assigned to the device to distinguish it from identical devices.
DEVICE.n.MODEL	O			String	Model number of the device.
DEVICE.n.SERIAL	O			String	Serial number of the device.
DEVICE.n.VERSION	O			String	Version of the firmware operating within the device.
DEVICE.n.GROUP	O			String	Name of group associated with
DEVICE.n.ROLE	O			String	Role the device has, if known
DEVICE.n.STATUS	R	Value	Description	I16	Indicates the condition of the device being reported. The reporting component may choose the condition level based on its own criteria.
		0	Debug		
		1	Normal / Green		
		2	Yellow		
		3	Orange		
		4	Red		
DEVICE.n.INFO	O			I16	An additional status code that can be supplied that is specific to that device.
DEVICE.n.NUM-OF-PARAMS	O	“n” starts at 1		U16	Number of additional parameters being reported that are associated with the device.
DEVICE.n.PARAM.n.NAME	O			String	Name of the additional parameter.
DEVICE.n.PARAM.n.TIME	O			Time	Time of parameter sampling
DEVICE.n.PARAM.n.VALUE	O			Variable	Value of the named parameter being reported. Component must ascertain the data type before accessing the value (e.g. with a function call).

The **Device** C2CX message is used to report the status of devices, physical or virtual. (The HEARTBEAT message and the CFG message are used to report status on software components.) The Device message would typically be used to report the status of devices that would not be capable of reporting themselves. For example, a software component may interact with a specialized device or

merely have access to the status of devices operating in the same environment. A designated software component would gather the status of the device(s) and publish the information with the Device C2CX message. This message is not intended to communicate with the device; that is, request the status of, or configure the device.

Thus, in conjunction with the CFG and Heartbeat messages, a full story on the configuration can be gathered for reporting and subsequent action-taking when a system-wide re-configuration is implemented.

Of course, this message does not need to be restricted to physical devices. Virtual devices may be constructed and reported on as well. A physical device may be logically partitioned, or a logical device may be spread over a number of physical devices. Or, a virtual (or pseudo) device could be constructed or defined with no relation to any physical device. For example, a set of parameters, somehow related, could be grouped as a “device” and reported on for display and monitoring. A single reporting agent could be responsible for a virtual device and report on it. Or, a number of agents could report on separate parameters and the collector of the DEV messages could effectively construct a virtual device from the disparate information. A set of key or critical parameters could be constructed and reported on using this method.

A hypothetical example for a communications data path could consist of a ground antenna, a ground station processor/controller, a data link, and a front-end processor. Together these devices could constitute a virtual data link device whose individual device status are collected (and logically ‘AND’ed together) to provide a GO/NOGO or Red/Yellow/Green status on the data link.

5.2.2.2.4 Component-To-Component Transfer Heartbeat Message

Table 5-45. Component-To-Component Transfer Message Contents for Heartbeat

Field Name	Req/Opt	Value/Description		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	Unique ID used to distinguish the message
Content Body Segment					
Body Content Subtype					
C2CX-SUBTYPE	R	Value	Description	Header String	Identifies the type of information being transferred between the Components
		HB	Heartbeat		
Heartbeat Rate Information					
COUNTER	O	1+		U16*	Indicates the number of times that the C2CX message heartbeat message has been published, including this message.
PUB-RATE	O			U16	Indicates the rate, in number of seconds, which the C2CX heartbeat message is being published by the component. A rate of zero or less indicates that this C2CX message is not repeatedly published by the component. The default publishing rate of the C2CX heartbeat message is 30 seconds.
Component Status Information					
COMPONENT-STATUS	O	Value	Description	I16	Indicates the condition of the component being monitored, typically itself, although it may be a proxy for a remote component. The component may choose the condition level based on its own criteria.
		0	Debug		
		1	Normal / Green		
		2	Yellow		
		3	Orange		
		4	Red		
COMPONENT-INFO	O			I16	An additional status code the component can supply that is specific to that component.
COMPONENT-INFO-DETAILS	O			String	Allows a component to detail its status in a verbose message
SW-VERSION	O			Variable	Version number identifier of the reporting component. Component must ascertain the data type before accessing the value (e.g. with a function call).
Component Resource Utilization Snapshot					
CPU-MEM	O	In megabytes		F32	Amount of memory being used at this time by this component.
CPU-UTIL	O			F32	Percentage of CPU being utilized.

* Note: At a rate of 2 messages per minute, this counter will overflow after 11 days.

2013: The SW-VERSION field data type was changed to “Variable” to accommodate the various ways of representing the version ID of an component, such as float and string. Before referencing the SW-VERSION, the user must ascertain the data type (e.g. with a function call).

The **Heartbeat** message is used to notify other components that the sending / publishing component is alive or active on the Information Bus. Other components monitoring the Heartbeat messages can determine what action to take, if any, when a Heartbeat message fails to appear as scheduled, or if the COMPONENT-STATUS is not normal (or green). If the component does not publish the heartbeat at the default rate, it can supply the publishing rate (in the PUB-RATE field) and a counter (COUNTER field) for a monitor to calculate when a heartbeat is expected or might be missing or late. Each component or system or mission can determine its own preferred heartbeat rate.

Not Using the COMPONENT-STATUS Field: If the COMPONENT-STATUS field is not to be used, then if the component is running but not 100%, then cease publishing the Heartbeat message. The termination of the Heartbeat message will then make auto/re-configuration options possible. Which is to say, if the component is either 100% or 0%, or those are the only two states the component can report, then using the COMPONENT-STATUS field is not necessary, as long as the component can cease publishing the Heartbeat message in circumstances when it knows it is not 100%.

Using the COMPONENT-STATUS Field: A component may typically only supply the COMPONENT-STATUS of 1 – Normal – Green. However, when the status of the component is less than normal / green (100%), say yellow (75%), indicating a less than optimal operating state, it may also supply a status code in the COMPONENT-INFO field. This code would only have context within that component. A component may also issue a Log Message in conjunction with a change in COMPONENT-STATUS. The component would include the COMPONENT-INFO value in the subsequent Log Message so the Heartbeat message and the Log Message could be cross-referenced. Components can self-determine what constitutes a yellow, orange, or red state of processing. A component that ceases to send a heartbeat message will be presumed to be absent and in a red condition. If applicable, a component will then be susceptible to a pre-determined recovery action, including failover and restart. A monitoring agent can use the COMPONENT-STATUS value to color code a display of the component's status.

Memory Leaks: A recurring and nagging problem in software development is the presence of memory leaks - the failure to return unused memory to the operating system. Over time, this causes a process (task) to use up all its allotted memory and/or the total system memory resulting in a bogged down or inoperable system. Having the component additionally report its own memory usage in the Heartbeat message could be used to monitor for and discover memory leaks during execution. A separate monitoring agent could collect the Heartbeat messages, and among other responsibilities, monitor and track the memory usage for each process over time. If the memory usage is detected to be steadily increasing over time, an action could be generated to issue a warning (Log) message. (An algorithm must be developed or found that would detect

memory leaks for any process, regardless of the amount of memory it may require on any operating system.)

5.2.2.2.5 Component-To-Component Transfer Resource Message

Table 5-46. Component-To-Component Transfer Message Contents for Resource

Field Name	Req/Opt	Value/Description		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	Unique ID used to distinguish the message
Content Body Segment					
Body Content Subtype					
C2CX-SUBTYPE	R	Value	Description	Header String	Identifies the type of information being transferred between the Components
		RSRC	Resource		
Resource Rate Information					
COUNTER	O	1+		U16*	Indicates the number of times that the C2CX Resource message has been published, including this message.
PUB-RATE	O	Seconds		U16	Rate the data is being collected and published. The default publishing rate is 30 seconds. A rate of zero or less indicates this message is not being repeatedly published.
Computer Processor Information					
OPER-SYS	O			String	Operating system component is using
NUM-OF-CPUS	O	1+		U16	Number of CPUs being monitored
CPU.n.MEM	O	In megabytes. "n" starts at 1		U32	Amount of memory for this CPU
CPU.n.MEM-UTIL	O	0-100		F32	Memory utilization. Percentage of memory utilized.
CPU.n.UTIL	O	0-100		F32	CPU utilization. Percentage of CPU utilized
CPU.n.PAGE-FAULTS	O	1+		U32	Number of page faults
Disk Information					
NUM-OF-DISKS	O	0+		U16	Number of disks being monitored
DISK.n.NAME	O	"n" starts at "1"		String	Name of the disk
DISK.n.SIZE	O	In megabytes		U32	Absolute size of the disk
DISK.n.UTIL	O	0-100		F32	Disk space utilization. Percentage of Disk space utilized.
Main Memory Information					
MEM.UTIL	O	0-100		F32	Percent of main memory utilized
MEM.PHYSICAL.TOTAL	O	1+		U64	Total amount of physical memory present, in bytes
MEM.PHYSICAL.AVAIL	O	0+		U64	Total amount of physical memory available, in bytes
MEM.VIRTUAL.TOTAL	O	1+		U64	Total amount of virtual memory present, in bytes
MEM.VIRTUAL.AVAIL	O	0+		U64	Total amount of virtual memory available, in bytes
Network Interface Information					
NUM-OF-NET-PORTS	O	1+		U16	Number of network ports
NET-PORT.n.NAME	O	"n" starts at "1"		String	Name of the network port
NET-PORT.n.EUI-ADR	O	Format of: 01-23-45-67-89-ab or 01:23:45:67:89:ab		String	Media Access Control (MAC) or Extended Unique Identifier (EUI) physical address. MAC-48, EUI-48, or EUI-64 format.
NET-PORT.n.IP-ADR	O	208.77.188.166 or 2001:0db8:85a3:08d3:1319:8a2e:0370:7334		String	Internet Protocol (IP) logical address. IPv4 (32-bit) or IPv6 (128-bit) format.

NET-PORT.n.TOTAL-BANDWIDTH	O	0+	U32	Bandwidth of the port in Kbps
NET-PORT.n.UTIL	O	0-100	F32	Percentage of Network port utilization
NET-PORT.n.BYTES-SENT	O	0+	U64	Number of bytes sent over the port
NET-PORT.n.BYTES-RECEIVED	O	0+	U64	Number of bytes received over the port
NET-PORT.n.ERRORS	O	0+	U32	Number of errors encountered on the port

* Note: At a rate of 2 messages per minute, this counter will overflow after 11 days.

The C2CX **Resource** message is used to publish computer performance data. Resource data is organized per CPU, disk, and network port. It is intended that the data be a snapshot of the resources at the time of collection and not a cumulative summary. The snapshot of the resources can be paired with the time of publication of the message (found in the Information Bus Header) to produce a data point. After the collection of a number of data points, a trend /plot of the resources can be established.

All resource information has been marked as optional so that a component may provide any or all of the resource information as necessary. For example, an agent collecting and publishing data may determine to publish the CPU resources at a difference rate than the disk resources. Resource messages for CPUs might be published every 60 seconds, while disk Resource messages might be published every 300 seconds.

If a component desires to be controlled or directed as to the frequency of resource publishing, it could use the C2CX message of C2CX-SUBTYPE=CNTRL with a CNTRL-STRING of "SET RSRC CPU 60" to set the CPU resources publication rate at 60 seconds (or disk resources at 300 seconds with "SET RSRC DISK 300").

5.2.2.2.3 Component-to-Component Transfer Message Subjects

Table 5-47. Component-to-Component Transfer Message Subject Naming

	Subject Standard	Mission Elements		Message Elements		<i>Miscellaneous Elements</i>		
Subject Element	Specification	MISSION	SAT	TYP	SUBTYP	ME1	ME2	ME3 ...
Subject Content	GMSEC	[mission]	[sat]	MSG	C2CX	[Component name of publisher]	[C2CX-Subtype: CFG, CNTL, DEV, HB, RSRC]	[Component name of recipient(s)]
Example for Publisher / Sender	GMSEC	MSSN	SAT1	MSG	C2CX	APP1	HB	
Example for Publisher / Sender	GMSEC	MSSN	SAT1	MSG	C2CX	CFGMGR	CNTL	AGENT3
Example for Subscriber / Receiver	GMSEC	MSSN	SAT1	MSG	C2CX	*	HB	>

Table 5-48. Properties of the *Miscellaneous Elements* for the Component-to-Component Transfer Messages

<i>Miscellaneous Element</i>	Required / Optional	Description	Field in Msg, if applicable
ME1	Required	Component name of Publisher	"COMPONENT" from Bus Header of msg
ME2	Required	[CFG, CNTL, DEV, HB, RSRC]	"C2CX-SUBTYPE" from msg content
ME3 ME4 ME5	Optional	Component destination: The three <i>miscellaneous elements</i> may be used to direct the message to a specific destination, as necessary, in Header String format. ME3 = destination component or group ME4 = destination node ME5 = destination facility	NA
ME6	Optional	A keyword the receiving process could use for filtering	"CNTL-KEYWORD" from msg content

Heartbeat messages are expected to be published to the general public and processed by any number of components. For this message only the *ME1* (component name of the publisher) and *ME2* (HB) elements are needed in the message subject.

The Configuration, Device, and Resource messages may also be published for the general public or they may be targeted to a central collector component. The third element, *ME3* (component of recipient), might be used, as necessary.

The Control messages may have the greatest need to be targeted to specific components. In this case, the *ME3*, *ME4*, and *ME5* elements would most likely be used. For example, a configuration may have a number of CPU(s) with identically named components. Perhaps a collector of resource information on one node communicates with a number of identically named agents spread over a group of computers. The collector may want to send C2CX CNTL messages to all the agents at once or to an individual agent in the group. To send a message to a single agent, it may be necessary to specify the targeted facility, computer node, and agent's component name. Or, just specify the computer node and component name. Or, even the computer node, component name, and subcomponent name. *Miscellaneous elements ME3 and beyond* can be used for this purpose to uniquely specify a destination.

Examples:

Publishing / Sending Configuration Status Control, Device, Heartbeat, and Resource messages:

GMSEC.MSSN.SAT1.MSG.C2CX.APP1.CFG

GMSEC.MSSN.SAT1.MSG.C2CX.APP1.CNTL or
GMSEC.MSSN.SAT1.MSG.C2CX.APP1.CNTL.COMPONENT or
GMSEC.MSSN.SAT1.MSG.C2CX.APP1.CNTL.COMPONENT.NODE.FACILITY or
GMSEC.MSSN.SAT1.MSG.C2CX.APP1.CNTL.COMPONENT.NODE.FACILITY.KEYWORD

GMSEC.MSSN.SAT1.MSG.C2CX.TLM3.HB

GMSEC.MSSN.SAT1.MSG.C2CX.AGENT23.RSRC or
GMSEC.MSSN.SAT1.MSG.C2CX.AGENT23.RSRC.COLLECTOR

Subscribing / Receiving a Configuration, Control, Device, Heartbeat, and Resource message:

(... MSG.C2CX.PUBLISHER.C2CX-SUBTYPE.[COMPONENT.NODE.FACILITY.KEYWORD])

GMSEC.*.*.MSG.C2CX.MYAGENT.CFG.CFGMGR.>
GMSEC.*.*.MSG.C2CX.CFGMGR.CNTL.>
GMSEC.*.*.MSG.C2CX.CFGMGR.CNTL.RSRCAGENTS.NODE-X.>
GMSEC.*.*.MSG.C2CX.CFGMGR.CNTL.CPUAGENTS.NODE-Y.>
GMSEC.*.*.MSG.C2CX.CFGMGR.CNTL.CPUAGENTS.NODE-Y.DO_ABC

GMSEC.*.*.MSG.C2CX.*.HB.>

GMSEC.*.*.MSG.C2CX.MYAGENT.RSRC.>

5.3 Data Level Messages

5.3.1 Telemetry Data Messages

Telemetry data can be packaged in a number of forms usually driven by the needs of the users. It can be “as is” as originally received from the spacecraft, but generally, once on the ground, practically all users prefer it to be cleaned and converted. That is to say,

- Cleaned – removal of downlink artifacts including duplication, applying some bit error correction, and ensuring the data is sequenced in time order (sorted in the order collected on the spacecraft)
- Converted – raw analog data is converted to a value with units. Other corrections, calibrations, and correlations may be applied that pertain to the instrument, the measurement, space and atmospheric physics, and so on.

The above is a simplified description and does not account for all processing applied to the data. The GMSEC architecture leaves the processing of the data to the domain experts and only provides the means for transporting the data, whatever state or condition it may be in.

GMSEC has defined and grouped a number of messages that allow for the delivery of telemetry with reference to time, format, and data selection.

- Time – delivery of data can be in real-time as it is received from the downlink, or it can be retrieved from an archive and transported again
- Format and data selection – data can be bundled in TDM/CCSDS formats of raw frames or packets, or packaged as selected mnemonics, raw and/or converted

Depending on the user’s needs for telemetry data the following table summarizes what messages should be used for the kind of data desired.

Table 5-49. Telemetry Data Messages and How They Are Used

		Time Reference of Data	
		Real-Time	Historical
Amount of Data	All	Real-Time Telemetry Data Messages	Replay Telemetry Data Messages
	Selected	Real-Time Mnemonic Value Messages	Archive Mnemonic Value Messages

When **All** historical data is desired, it can be bounded or restricted further by specifying

- Time frame (start and stop times)
- Kind of data (real data, simulator data, test data)
- Data format (TDM frames, CCSDS packets, CCSDS frames)
- Channel IDs and AP IDs

When **Selected** historical data is desired, the requestor must also specify

- The mnemonics
- Sampling frequency
 - All samples
 - Upon change
 - By time period

Further details of the telemetry data messages immediately follow.

5.3.1.1 Real-Time Telemetry Data Messages

Telemetry Messages are data packages that contain spacecraft health and safety data. In most ground systems, a Telemetry Message is packaged by the spacecraft and sent to the ground station. The ground station performs air-to-ground quality checking, adds ground station information, and routes the data to the ground system. Archive retrieval systems, simulators, and data generators can also provide Telemetry Messages in replay, simulation, and test modes.

Additionally, the data can be published “as is” (Raw), or after a degree or level of processing (Processed). The latter could involve a number of data scrubbing techniques plus conversion from binary values to engineering units. The following table lists the messages that have been defined to transport the various kinds of telemetry data.

Table 5-50. Telemetry Messages

Telemetry Message	Data Form (Level)	Data Format
Telemetry CCSDS Packet	Raw	CCSDS Packet
Telemetry CCSDS Frame	Raw	CCSDS Frame
Telemetry TDM Frame	Raw	TDM Frame
Processed Telemetry Data	Processed (Converted)	Data samples for one frame organized by mnemonic

The CCSDS Frame and CCSDS Packet are industry standard formats. Time Division Multiplexing (TDM) is the method and format for sending multiple digital signals along a single telecommunications transmission. Specific decommutation instructions for the frames and packets are documented in other resources.

Table 5-51. Telemetry Message Summary

Sender	A GMSEC compliant application such as a ground station, simulator, archive component. Or front-end processor
Senders Intended Usage	Publish
Receiver	Telemetry Decommutation System, Archive System, Trending System, Expert System
Receivers Intended Usage	Subscribe
What	Spacecraft health and safety data to be decommutated and/or archived
When	As needed but usually dependent on data rate and/or replay rate
Quality of Service	Reliable

Example:

1. Spacecraft health and safety data sent from ground station to ground system

5.3.1.1.1 Real-Time Telemetry Data Messages Information Bus Header

The abbreviated table below shows the required Values of the MESSAGE-TYPE and MESSAGE-SUBTYPE fields for this message.

Table 5-52. Telemetry Message Information Bus Header

Field Name	Req/ Opt/API	Value	Type	Notes
Message Information				
HEADER-VERSION	R	2010	F32	Version Number for this message description.
MESSAGE-TYPE	R	MSG	Header string	Message type identifier: REQ, RESP, or MSG
MESSAGE-SUBTYPE	R	TLM	Header string	Unique message identifier, fixed for GMSEC Standard Messages
More ... Please refer to Table 4-9. GMSEC Information Bus Header for a complete definition.

5.3.1.1.2 Real-Time Telemetry Data Message Contents

This Section consists of the Telemetry Messages contents for the CCSDS Packet, CCSDS Frame, TDM format, and the Processed Telemetry format. Additional telemetry message contents may be added as necessary.

5.3.1.1.2.1 Telemetry Message Contents for CCSDS Packet

The Telemetry Message Contents for a CCSDS Packet is used for transferring CCSDS telemetry packets within the GMSEC architecture. The FORMAT field of the Telemetry Message Contents is set to CCSDSPKT. The Telemetry Message Contents consists of the raw CCSDS telemetry packet, time of the packet and the quality of the data. The STREAM-MODE field is used to classify the kind or source of the CCSDS packets. The mode of the telemetry data can be either real-time (RT), replay (RPY), simulation (SIM), or test (TEST).

Table 5-53. Telemetry Message Contents for CCSDS Packet

Field Name	Req/ Opt	Value		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	Unique ID used to distinguish the message
Content Body Segment					
Structure: Telemetry Data Stream Information					
FORMAT	R	CCSDSPKT		String	Message contains CCSDS packet
COLLECTION-POINT	O			String	Receiver, device, point, path, etc. where data was received. Used to distinguish data simultaneously received at multiple collection points.
STREAM-MODE	R	Value	Description	String	Identifies the mode of the stream of telemetry as either Real-time, Replay, Simulator, or Test.
		RT	Real-time		
		RPY	Replay		
		SIM	Simulator		
FINAL-MESSAGE	O	Value	Description	Boolean	When true (and known, especially for replay data), indicates the last message in the stream.
		0	No/False		
		1	Yes/True		
TIME	O			Time	Time of packet, usually ground receipt time
Structure: Telemetry Data Channel Information					
PHY-CHAN	R			String	Physical channel on which data is received
VCID	R			I16	Virtual channel ID
QUALITY-CHECK	O	Value	Description	I16	Indicates quality checking was performed for reason indicated.
		Bit 0	Partial Packet		
QUALITY	O	Value	Description	I16	Indicates quality state if checking was performed
		Bit 0	Partial Packet		
DATA	R			Binary (Blob)	Raw telemetry data

5.3.1.1.2.2 Telemetry Message Contents for CCSDS Frame

The Telemetry Message Contents for a CCSDS Frame is used to transfer CCSDS telemetry frames within the GMSEC architecture. The FORMAT field of the Telemetry Message Contents is set to CCSDSFRAME. The Telemetry Message Contents consists of the raw CCSDS telemetry frame, time of the frame, quality checking performed, and the resulting quality of the data. The STREAM-MODE field is used to classify the kind or source of the CCSDS frame.

A frame with a Frame Sync pattern at the front that includes Reed-Solomon check symbols is called a Coded Virtual Channel Data Unit (CVCDU) in the CCSDS documentation.

Table 5-54. Telemetry Message Contents for CCSDS Frame

Field Name	Req/Opt	Value/Description		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	Unique ID used to distinguish the message
Content Body Segment					
Structure: Telemetry Data Stream Information					
FORMAT	R	CCSDSFRAME		String	Message contains a CCSDS Frame
COLLECTION-POINT	O			String	Receiver, device, point, path, etc. where data was received. Used to distinguish data simultaneously received at multiple collection points.
STREAM-MODE	R	Value	Description	String	Identifies the kind or source of the stream of telemetry as either Real-time, Replay, Simulator, or Test.
		RT	Real-time		
		RPY	Replay		
		SIM	Simulator		
		TEST	Test/Data Generator		
FINAL-MESSAGE	O	Value	Description	Boolean	When true (and known, especially for replay data), indicates the last message in the stream.
		0	No/False		
		1	Yes/True		
LENGTH	O	Bytes		U32	Length of frame
TIME	O			Time	Time of frame, usually ground receipt time
Structure: Telemetry Data Channel Information					
PHY-CHAN	R			String	Physical channel on which data is received
VCID	R			I16	Virtual Channel ID
Telemetry Frame Quality Information					
FRAMESYNC-STATUS	O	Value	Description	I16	State of frame synchronization from equipment when frame is ingested
		1	Search		
		2	Verify		
		3	Lock		
		4	Check		
RS-PRESENT	O	Value	Description	Boolean	Indicates if the Reed-Solomon codes are present in the data.
		0	No/False		
		1	Yes/True		
QUALITY-CHECK	O	Value	Description	I16	Indicates quality checking performed, if applicable. If the bit is set the particular quality check was performed.
		Bit 0	CRC Quality Check		

		Bit 1	Reed-Solomon Quality Check		
		Bit 2	Turbo Code Quality Check		
QUALITY	O	Value	Description	116	Indicates quality state if checking is performed. If the bit is set the particular quality check failed.
		Bit 0	CRC Quality State		
		Bit 1	Reed-Solomon Quality State		
		Bit 2	Turbo Code Quality State		
Telemetry Data					
DATA	O			Binary (Blob)	Raw telemetry data

5.3.1.1.2.3 Telemetry Message Contents for TDM Data

The Telemetry Message Contents for a Time-Division Multiplexing (TDM) frame is used to transfer TDM data within the GMSEC architecture. The FORMAT field of the Telemetry Message Contents is set to TDM. The Telemetry Message Contents simply consists of the raw TDM frame, length of the frame, and time of the frame. The STREAM-MODE field is used to classify the kind or source of the TDM frame.

Table 5-55. Telemetry Message Contents for TDM Frame

Field Name	Req/ Opt	Value/Description		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	Unique ID used to distinguish the message
Content Body Segment					
Structure: Telemetry Data Stream Information					
FORMAT	R	TDM		String	Message contains a TDM Frame
COLLECTION-POINT	O			String	Receiver, device, point, path, etc. where data was received. Used to distinguish data simultaneously received at multiple collection points.
STREAM-MODE	R	Value	Description	String	Identifies the kind or source of the stream of telemetry as either Real-time, Replay, Simulator, or Test.
		RT	Real-time		
		RPY	Replay		
		SIM	Simulator		
FINAL-MESSAGE	O	Value	Description	Boolean	When true (and known, especially for replay data), indicates the last message in the stream.
		0	No/False		
		1	Yes/True		
LENGTH	O	Bytes		U32	Length of frame
TIME	O			Time	Time of frame, usually ground receipt time
Telemetry Frame Quality Information					
FRAMESYNC-STATUS	O	Value	Description	I16	State of frame synchronization from equipment when frame is ingested
		1	Search		
		2	Verify		
		3	Lock		
		4	Check		
Telemetry Data					
DATA	O			Binary (Blob)	Raw telemetry data

5.3.1.1.2.4 Telemetry Message Contents for Processed Telemetry Frame

The Processed Telemetry Message is a hybrid between the unprocessed (raw) Telemetry Message and the Mnemonic Value Data Message. It contains both raw and converted data for a frame of telemetry data that is organized in the message by mnemonic. Thus, it is frame-based as are the telemetry messages, but mnemonic-organized as are the Mnemonic Value Data Messages. It serves to provide all the telemetry data in a raw and processed format. Therefore, many consumers could be provided with a substantial amount of data without needing to specifically request a custom selected mnemonic data set.

When this message is published is to be determined by the mission or data provider. It could be published “alongside” or in accordance with the raw telemetry data messages, or by itself. Also, it could be published automatically or only by request (as a replay).

The FORMAT field of the Processed Telemetry Message Contents is set to PROCESSED. The STREAM-MODE field is used to classify the kind or source of the frames (or packets). The mode of the telemetry data can be either real-time (RT), replay (RPY), simulation (SIM), or test (TEST).

Table 5-56. Telemetry Message Contents for a Processed Telemetry Frame

Field Name	Req/ Opt	Value/Description		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	Unique ID used to distinguish the message
Content Body Segment					
Structure: Telemetry Data Stream Information					
FORMAT	R	PROCESSED		String	Message contains a processed frame
COLLECTION-POINT	O			String	Receiver, device, point, path, etc. where data was received. Used to distinguish data simultaneously received at multiple collection points.
STREAM-MODE	R	Value	Description	String	Identifies the kind or source of the stream of telemetry as either Real-time, Replay, Simulator, or Test.
		RT	Real-time		
		RPY	Replay		
		SIM	Simulator		
		TEST	Test/Data Generator		
FINAL-MESSAGE	O	Value	Description	Boolean	When true (and known, especially for replay data), indicates the last message in the stream.
		0	No/False		
		1	Yes/True		
LENGTH	O	Bytes		U32	Length of frame
TIME	O			Time	Time of frame, usually ground receipt time
Telemetry Frame Quality Information					
FRAMESYNC-STATUS	O	Value	Description	I16	State of frame synchronization from equipment when frame is ingested
		1	Search		
		2	Verify		
		3	Lock		

		4	Check		
Telemetry Data Description					
NUM-OF-FORMAT-IDENTIFIERS	R	0+	U16	Number of fields used to identify the frames (e.g. TDM major/minor frames would have a value of 2). Zero is only permissible for vehicles with one telemetry format with a single type of frame.	
FORMAT-IDENTIFIER.1.VALUE	D		String	Value of the first field used to identify the telemetry format (e.g. value of major frame ID for TDM telemetry). If the message is used with XTCE, this is the first restriction criteria in an XTCE container.	
FORMAT-IDENTIFIER.n.VALUE	D		String	Value of the nth field used to identify the telemetry. If the message is used with XTCE, this is the last restriction criteria in an XTCE container.	
Common Information for All Mnemonics					
NUM-OF-MNEMONICS	R	1+	U16	Total number of mnemonics in this message	
Common Information for a Single Mnemonic					
MNEMONIC.1.NAME	R		String	Name of the first mnemonic	
MNEMONIC.1.STATUS	R	Value	Description	I16	Status of the first mnemonic: valid mnemonic, or valid mnemonic with no data, or invalid mnemonic
		1	Valid		
		2	Valid, No data		
		3	Invalid		
MNEMONIC.1.UNITS	O		String	Units associated with the value converted to engineering units for the first mnemonic	
MNEMONIC.1.NUM-OF-SAMPLES	R		U16	Number of data samples for the first mnemonic. This value should equal the number of times the mnemonic appears in the telemetry frame (e.g. will be greater than 1 for super-commutated telemetry points).	
First Single Data Point Information for the First Mnemonic					
MNEMONIC.1.SAMPLE.1.TIME-STAMP	O		Time	Time stamp for the first data sample of the first mnemonic	
MNEMONIC.1.SAMPLE.1.RAW-VALUE	O		I32	Raw value for the first data sample of the first mnemonic	
MNEMONIC.1.SAMPLE.1.EU-VALUE	O		F32	Raw value converted to Engineering Units if engineering units conversion is present for the first data sample of the first mnemonic	
MNEMONIC.1.SAMPLE.1.TEXT-VALUE	O		String	Raw value converted to a text string if text conversion is present for the first data sample of the first mnemonic	
MNEMONIC.1.SAMPLE.1.LIMIT-ENABLE-DISABLE	R	Value	Description	Boolean	Indicates the limit checking state for the first data sample of the first mnemonic
		0	Disabled		
		1	Enabled		
: : : : : : : :	O	: : : : : : : : : : : : : : : : : :	: : : : : : : :	: : : : : : : : : : : : : : : : : :	
Last Single Data Point Information for the First Mnemonic					
MNEMONIC.1.SAMPLE.n.TIME-STAMP	O		Time	Time stamp for the 'n th ' data sample of the first mnemonic	
MNEMONIC.1.SAMPLE.n.RAW-VALUE	O		I32	Raw value for the 'n th ' data sample of the first mnemonic	
MNEMONIC.1.SAMPLE.n.EU-VALUE	O		F32	Raw value converted to Engineering Units if engineering units conversion is	

					present for the 'n th ' data sample of the first mnemonic
MNEMONIC.1.SAMPLE .n.TEXT-VALUE	O			String	Raw value converted to a text string if text conversion is present for the 'n th ' data sample of the first mnemonic
MNEMONIC.1.SAMPLE .n.LIMIT-ENABLE- DISABLE	R	Value	Description	Boolean	Indicates the limit checking state for the 'n th ' data sample of the first mnemonic
		0	Disabled		
		1	Enabled		
: : : : : :	:::	: : : : : : : : : : : :		: : : : : : : :	: : : : : : : : : : ~ :
Common Information for the Last Mnemonic					
MNEMONIC.n.NAME	R			String	Name of the 'n th ' mnemonic
MNEMONIC.n.STATUS	R	Value	Description	I16	Status of the 'n th ' mnemonic: valid mnemonic, or valid mnemonic with no data, or invalid mnemonic
		1	Valid		
		2	Valid, Nodata		
		3	Invalid		
MNEMONIC.n.UNITS	O			String	Units associated with the raw value converted to engineering units for the 'n th ' mnemonic
MNEMONIC.n.NUM-OF-SAMPLES	R			U16	Number of data samples for the 'n th ' mnemonic. This value should equal the number of times the mnemonic appears in the telemetry frame (e.g. will be greater than 1 for super-commutated telemetry points).
First Single Data Point Information for the Last Mnemonic					
MNEMONIC.n.SAMPLE .1.TIME-STAMP	O			Time	Time stamp for the first data sample of the 'n th ' mnemonic
MNEMONIC.n.SAMPLE .1.RAW-VALUE	O			I32	Raw value for the first data sample of the 'n th ' mnemonic
MNEMONIC.n.SAMPLE .1.EU-VALUE	O			F32	Raw value converted to Engineering Units if engineering units conversion is present for the first data sample of the 'n th ' mnemonic
MNEMONIC.n.SAMPLE .1.TEXT-VALUE	O			String	Raw value converted to a text string if text conversion is present for the first data sample of the 'n th ' mnemonic
MNEMONIC.n.SAMPLE .1.LIMIT-ENABLE- DISABLE	R	Value	Description	Boolean	Indicates the limit checking state for the first data sample of the 'n th ' mnemonic
		0	Disabled		
		1	Enabled		
: : : : : :	:::	: : : : ~ : : : : : : : : : : :		: : : : ~ : : : : : : : :	: : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~ : : : : ~

5.3.1.1.3 Real-Time Telemetry Data Message Subjects

Table 5-57. Telemetry Message Subject Naming

	Subject Standard	Mission Elements		Message Elements		Miscellaneous Elements				
Subject Element	Specification	MISSION	SAT	TYP	SUBTYP	ME1	ME2	ME3	ME4	ME5
Subject Content	GMSEC	[mission]	[sat]	MSG	TLM	Component of publisher	Stream-mode	Format	Virtual Channel ID	AP ID
Example for Publisher / Sender	GMSEC	MSSN	SAT1	MSG	TLM	SATSIM	SIM	CCSDSFRA ME	2	1
Example for Publisher / Sender	GMSEC	MSSN	SAT1	MSG	TLM	TFEP	RT	CCSDSPKT	1	2
Example for Subscriber / Receiver	GMSEC	*	SAT1	MSG	TLM	*	RT	*	*	*

Table 5-58. Properties of the *Miscellaneous Elements* for the Telemetry Message

Miscellaneous Element	Required / Optional	Description	Field in Msg, if applicable
ME1	Required	Component name of Publisher	"COMPONENT" from Bus Header of msg
ME2	Required	Identifies stream as real-time, playback, simulator, or test.	'STREAM-MODE" from msg content
ME3	Required	Telemetry format	'FORMAT" from the msg content
ME4	Required for all CCSDS	Virtual channel ID	"VCID" from msg content, or from header portion of CCSDS frame
ME5	Required for CCSDS Packet only	AP ID – identifies a particular subsystem on the spacecraft	From header portion of data stream
ME6	Optional	Point on ground system where data was captured	'COLLECTION-POINT" from msg content

Example for Publisher / Sender of Telemetry Messages:

GMSEC.MSSN.SAT1.MSG.TLM.SATSIM.SIM.CCSDSFRAME.2
 GMSEC.MSSN.SAT1.MSG.TLM.SATSIM.SIM.CCSDSPKT.2.1
 GMSEC.MSSN.SAT1.MSG.TLM.TFEP.RT.TDM
 GMSEC.MSSN.SAT1.MSG.TLM.TFEP.RT.TDM.FILL.FILL.ANTENNA5
 GMSEC.MSSN.SAT1.MSG.TLM.DECOM.RT.PROCESSED
 GMSEC.MSSN.SAT1.MSG.TLM.DECOM.RT.PROCESSED.FILL.FILL.ANTENNA5

Example for Subscriber / Receiver of Telemetry Messages:

GMSEC.MSSN.*.MSG.TLM.*.SIM.CCSDSFRAME.>

GMSEC.MSSN.*.MSG.TLM.TFEP.RT.CCSDSPKT.>
GMSEC.MSSN.SAT1.MSG.TLM.TFEP.RT.CCSDSPKT.2.1
GMSEC.MSSN.SAT1.MSG.TLM.DECOM.RT.TDM.*.ANTENNA5
GMSEC.MSSN.SAT1.MSG.TLM.DECOM.RT.PROCESSED.>

5.3.1.2 Replay Telemetry Data Messages

Note: Originally, the Replay (Request and Response) Telemetry Data Messages were designed for historical, archived data only. **However, these messages can also be used to request real-time data or even future data.** Because not all downlinked data is automatically forwarded (published) in real-time, these messages also provide the means to request the publication of a current or future data stream in real-time.

The Replay Telemetry Data Messages provide access to streams of raw telemetry data (packets, frames, etc.) and also processed (converted) data. This data can be current in real-time, data replayed from a data archive, data from a simulator or data generator, or even a future data set. A request for future data occurs before the satellite has downlinked the data, and ensures that once the data is collected on the ground it will be forwarded in real-time.

The Replay Telemetry Data Messages are an example of the Request-Response-Message Triad. That is, the Replay Telemetry Data Request Message is followed by the Replay Telemetry Data Response Message followed by a series or stream of Telemetry Data messages.

A common use of the Replay Telemetry Data Messages is when a decommutation component needs to process raw archived telemetry data. The decommutation component builds a Replay Telemetry Data Request, specifying the source of telemetry data, range of data, and the playback speed. The component sends the request to a Telemetry Archive component. The Telemetry Archive component processes the request by locating the requested telemetry data from the archive and returning the status of the request in a Replay Telemetry Data Response Message. The Telemetry Archive component will then retrieve the telemetry data from the archive and publish it in Replay Telemetry Data Messages at the requested speed. The requesting component (who has subscribed to the Telemetry Data Messages) receives and processes the requested telemetry data. The decommutation component may in turn provide processed telemetry messages or archived mnemonic data value messages.

As stated above, requests for real-time data can be for current streaming data or for a future data stream, one not yet received at a ground station. Requests for real-time data present some new ways of thinking in light of present day technology. Though not applicable in the past, some present day recording devices can pause and resume real-time data streams, and even step through them. Traditionally, these features were only available or associated with playback data streams, but more sophisticated recording devices have merged these playback capabilities with real-time data streams. This has led to some new terms & concepts such as a “paused real-time data stream”, a “resumed real-time data stream”, and a “real-time data stream playing at half speed”.

Therefore, depending on the sophistication of the data provider and recording mechanism, some features traditionally associated with a playback data stream may be available with a real-time data stream. However, if the data provider does not provide such features, it must return an appropriate status to the requestor in the Response message.

5.3.1.2.1 Replay Telemetry Data Request Message

A Replay Telemetry Data Request Message is a service request that is issued to a telemetry data provider by an application that desires telemetry data. The request could be for archived or real-time data. For an archived data request, every field in the Replay Telemetry Data Request Message will be valid. That is, any field can be used. However, since some fields are mutually exclusive, not all fields will be used. The Replay Telemetry Response Message returns the status of the request. Please see Section 4.2 GMSEC Messages: Their Characteristics and Interactions for a general discussion on these types of messages.

For all requests, the mission and satellite identification is found in the message header (and associated message subject). Other data selection parameters include:

- Data stream characteristics (flow control, speed, data type)
- Broad data selection parameters (time window, orbit no., or data file name)
- Refined data selection parameters (data format and data specific)

For a real-time data stream request, the following fields of the Request message are not valid:

- PLAYBACK-RATIO*
- DATA-RATE*
- File name fields

* However, as mentioned in the previous section, a more sophisticated data provider may be able to stream real-time data at rates slower than real-time.

Table 5-59. Replay Telemetry Request Message Summary

Sender	Any GMSEC compliant application requesting archived telemetry data
Senders Intended Usage	Request
Receiver	Any GMSEC compliant application with access to a telemetry archive
Receivers Intended Usage	Subscribe
What	Telemetry data as Telemetry Messages
When	As needed
Quality of Service	Guaranteed

Examples:

1. Archived telemetry data replayed to a T&C component.
2. “Register” to receive a future real-time telemetry data stream.

5.3.1.2.1.1 Replay Telemetry Data Request Message Information Bus Header

The abbreviated table below shows the required values of the MESSAGE-TYPE and MESSAGE-SUBTYPE fields for this message.

Table 5-60. Replay Telemetry Request Message Information Bus Header

Field Name	Req/ Opt/API	Value	Type	Notes
Message Information				
HEADER-VERSION	R	2010	F32	Version Number for this message description.
MESSAGE-TYPE	R	REQ	Header string	Message type identifier: REQ, RESP, or MSG
MESSAGE-SUBTYPE	R	RTLTM	Header string	Unique message identifier, fixed for GMSEC Standard Messages
More ... Please refer to Table 4-9. GMSEC Information Bus Header for a complete definition.

5.3.1.2.1.2 Replay Telemetry Request Message Contents

Used for real-time and archived data.

Table 5-61. Replay Telemetry Request Message Contents

Field Name	Req/Opt	Value		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	Unique ID used to distinguish the message
Data Stream Characteristics					
ACTION	R	Value	Description	I16	Identifies the desired action to perform on the telemetry data stream.
		1	Start		
		2	Stop		
		3	Pause		
		4	Continue		
		5	Step		
STREAM-MODE	R	String	Description	String	Identifies the type of data to be published.
		RT	Real Time		
		RPY	Replay / Playback		
		SIM	Simulator		
		TEST	Test / Data Generator		
Data Stream Speed: (Choose One of Two.) Data Rate or Playback Ratio.					
PLAYBACK-RATIO	O	> 0 and < 1 is slower than real-time rate = 1 is equal to the real-time rate > 1 is faster than real-time rate		F32	Speed of playback as a ratio of playback rate to real-time rate. This is the default method; default = 1.
DATA-RATE	O	> 0		U16	Data rate in Kilobits per second
Gross Data Selection (Choose One Method of Three): by Time Window, Orbit number, or by File Name					
STRUCTURE: Time Window					
START-TIME	O			Time (absolute or relative)	Time of first telemetry data. Defaults to the start of the archive
STOP-TIME	O			Time (absolute or relative)	Time of last telemetry data. Defaults to the end of the archive
Revolution					
ORBIT	O	0+		U32	Orbit or revolution number of the vehicle (past or future).
File Names					
NUM-OF-FILES	O	1+		U16	Number of Telemetry files to replay
FILE.1.NAME-PATTERN	O			String	Name of the first file
.....	
FILE.n.NAME-PATTERN	O				Name of the last file
Specific Telemetry Data Selection / Filtering Criteria					
FORMAT	R	Value	Description	String	Telemetry Message types to playback. Note: A provider may not capable of providing all types, e.g. only raw or only processed data.
		ALL	All message types		
		CCSDSPKT	CCSDS packets		
		CCSDSFRAME	CCSDS frames		
		TDM	TDM frames		

		PROCESSED	Converted TLM		
COLLECTION-POINT	O			String	Receiver, device, point, path, etc. where data was received. Used to distinguish data simultaneously received at multiple collection points.
STREAM-MODE	R	String	Description	String	Identifies the type of data to be published.
		RT	Real Time		
		RPY	Replay / Playback		
		SIM	Simulator		
		TEST	Test / Data Generator		
VCID	O			String	Virtual Channel IDs: comma delimited channel IDs with '-' for channel ID ranges. Example: 1,2,6-9,10
APID	O			String	AP IDs: comma delimited AP IDs with '-' for channel ID ranges. Example: 1,2,6-9,10

For an explanation on how the START-TIME and STOP-TIME could operate, see Table 5-19. Examples of Start and Stop Times.

Field Name Usage:

ACTION:

This field controls the flow of the data from the provider. Once a telemetry data stream has begun, the requestor may Pause it, Continue it, Step through it (frame-by-frame or Packet-by-packet), or Stop it. If a data provider does not provide all the options of the data flow, it should respond with the appropriate status in a Reply Telemetry Data Response Message.

STREAM-MODE:

This field identifies the type or source of the desired data.

Data Stream Speed:

The requestor has two methods of specifying the replay speed of the selected telemetry data in the fields PLAYBACK-RATIO and DATA-RATE. Either method can be used, but not both. If both methods are specified in the request, the PLAYBACK-RATIO should default. DATA-RATE is the rate the data will be replayed in kilobits per second. PLAYBACK-RATIO is a ratio of the playback speed to the real-time speed. If the rate of the replay is to be the same as the real-time rate, the ratio would be REPLAY: REAL-TIME or 1. If the replay speed is to be twice as fast as the real-time rate, the ratio would be 2. If the replay speed is to be one-tenth the speed of the real-time rate, the ratio would be 0.1. Thus, the PLAYBACK-RATIO must be greater than 0. No upper bound is placed on the PLAYBACK-RATIO, however, the responder/publisher of the data may self-impose their own replay limit.

COLLECTION-POINT:

Some satellites may be in contact with more than one ground station or receiver at a time, with each simultaneously collecting the downlinked data stream. If a requestor desires a data stream from a specific collection point, this field can be used to specify that site.

Broad Data Selection Parameters:

The requestor of a telemetry data replay (or playback) can select the limits or range of the data to be replayed by specifying a

- time window,
- orbit number, or
- by naming specific files of data.

Any method can be used to limit the data, but not more than one. If Start and Stop times are present in the request message, this method will take precedence over any other provided information. Specifying specific data files will take precedence over orbit number.

Refined Data Selection Parameters (data format and data specific):

Requestors are required to specify the data format (FORMAT). More specific data selection can be accomplished with the virtual channel (VCID) and AP ID (APID) fields.

5.3.1.2.2 Replay Telemetry Data Response Message

A Replay Telemetry Data Response Message is sent by a telemetry data stream provider in response to a Replay Telemetry Data Request Message. The primary purpose of the Replay Telemetry Data Response Message is to provide acknowledgment of the Replay Telemetry Data Request Message and a status of the action completed. Multiple Replay Telemetry Data Response Messages may be desired by the requestor if the processing and completion of the request is lengthy. In this event, an interim or interactive “working” type message would be issued to let the original application know that the action is still being processed.

Table 5-62. Replay Telemetry Response Message Summary

Sender	Application that received the Replay Telemetry Request Message
Senders Intended Usage	Reply
Receiver	Application that issued the Replay Telemetry Request Message and an application collecting Messages for audit trail purposes
Receivers Intended Usage	Subscribe
What	Provide success/failure response to the service that was requested
When	Upon receipt of Replay Telemetry Request Message or on an interval for those services that are time-consuming
Quality of Service	Guaranteed

Example:

1. Previously recorded spacecraft health and safety data retrieved from an archive and sent to a Telemetry and Command System.

5.3.1.2.2.1 Replay Telemetry Data Response Message Information Bus Header

The abbreviated table below shows the required Values of the MESSAGE-TYPE and MESSAGE-SUBTYPE fields for this message.

Table 5-63. Replay Telemetry Response Information Bus Header

Field Name	Req/ Opt/API	Value	Type	Notes
Message Information				
HEADER-VERSION	R	2010	F32	Version Number for this message description.
MESSAGE-TYPE	R	RESP	Header string	Message type identifier: REQ, RESP, or MSG
MESSAGE-SUBTYPE	R	RTLTM	Header string	Unique message identifier, fixed for GMSEC Standard Messages
More ... Please refer to Table 4-9. GMSEC Information Bus Header for a complete definition.

5.3.1.2.2.2 Replay Telemetry Data Response Message Contents

Table 5-64. Replay Telemetry Response Message Contents

Field Name	Req/ Opt	Value	Type	Notes
STRUCTURE: Body Content Identification				
CONTENT-VERSION	R	2016	F32	Version Number for this message content description
UNIQUE-ID	A		Header String	.
STRUCTURE: Response Status				
RESPONSE-STATUS	R	Value	Description	Identifies the status of the request being processed
		1	Acknowledgement	
		2	Working/Keep Alive	
		3	Successful completion	
		4	Failed completion	
		5	Invalid Request	
		6	Final Message	
TIME-COMPLETED	O		Time	Time application completed processing the request
RETURN-VALUE	O		I32	Return value or status based on the RESPONSE-STATUS. Useful to provide function call status or error code in the case of failed completion
STRUCTURE: Data Abstract				
DATA	O		Dependent upon response	Additional data that may be desired along with the completion status

5.3.1.2.3 Replay Telemetry Data Message Subjects

Table 5-65. Replay Telemetry Request Message Subject Naming

	Subject Standard	Mission Elements		Message Elements		Miscellaneous Elements		
Subject Element	Specification	MISSION	SAT	TYP	SUBTYP	ME1	ME2	ME3
Subject Content	GMSEC	[mission]	[sat]	REQ	RTLTM	[Component: TLM3, TLM2...]		
Example for Publisher / Sender	GMSEC	MSSN	SAT1	REQ	RTLTM	TLM2		
Example for Publisher / Sender	GMSEC	MSSN	SAT1	REQ	RTLTM	TLM3		
Example for Subscriber / Receiver	GMSEC	*	SAT1	REQ	RTLTM	TLM3		

Table 5-66. Properties of the *Miscellaneous Elements* for the Replay Telemetry Request Message

Miscellaneous Element	Required / Optional	Description	Field in Msg, if applicable
ME1	Required	Component name of Responder	NA
ME2	Not used		
ME3	Not used		

Examples:

Two components, ARCHIVER and TLM3, interact with the Replay Telemetry Request Message.

TLM3 sends a message with the following subject to request a Replay of Telemetry.

GMSEC.MSSN.SAT1.REQ.RTLM.ARCHIVER

ARCHIVER subscribes to receive the Replay Telemetry Request Message.

GMSEC.MSSN.SAT1.REQ.RTLM.ARCHIVER or
GMSEC.*.*.REQ.RTLM.ARCHIVER

Table 5-67. Replay Telemetry Response Message Subject Naming

	Subject Standard	Mission Elements		Message Elements		Miscellaneous Elements		
Subject Element	Specification	MISSION	SAT	TYP	SUB TYP	ME1	ME2	ME3
Subject Content	GMSEC	[mission]	[sat]	RESP	RTLTM	[Component: TLM3, TLM2...]	[Response-Status: ack, working, success, failure]	[Response-Status: ack, working, success, failure]
Example for Publisher / Sender	GMSEC	MSSN	SAT1	RESP	RTLTM	MGR	1	1
Example for Publisher / Sender	GMSEC	MSSN	SAT1	RESP	RTLTM	MGR	3	3
Example for Subscriber / Receiver	GMSEC	*	SAT1	RESP	RTLTM	MGR	*	

Table 5-68. Properties of the *Miscellaneous Elements* for the Replay Telemetry Response Message

Miscellaneous Element	Required / Optional	Description	Field Origination in Msg, if applicable
ME1	Required	Component name of Requestor	Echo of "COMPONENT" in header of Request msg
ME2	Required	Status type supplied by Responder	"RESPONSE-STATUS" from content of Response message

Examples:

Two components, ARCHIVER and TLM3 interact with the Replay Telemetry Response Message.

ARCHIVER sends a Replay Telemetry Response Message back to the requestor.

GMSEC.MSSN.SAT1.RESP.RTLM.TLM3.1

The original requestor subscribes to the Replay Telemetry Response Message.

GMSEC.MSSN.SAT1.RESP.RTLM.TLM3.>

Table 5-69. (Replay) Telemetry Message Subject Naming

	Subject Standard	Mission Elements		Message Elements		Miscellaneous Elements				
Subject Element	Specification	MISSION	SAT	TYP	SUBTYP	ME1	ME2	ME3	ME4	ME5
Subject Content	GMSEC	[mission]	[sat]	MSG	TLM	[Component of publisher]	[Format]	[Virtual Channel ID]	[AP ID]	
Example for Publisher / Sender	GMSEC	MSSN	SAT1	MSG	TLM	SATSIM	CCSDSFRAME	2	1	
Example for Publisher / Sender	GMSEC	MSSN	SAT1	MSG	TLM	TFEP	CCSDSPKT	1	2	
Example for Subscriber / Receiver	GMSEC	*	SAT1	MSG	TLM	*	*	*	*	*

Table 5-70. Properties of the *Miscellaneous Elements* for the (Replay) Telemetry Message

Miscellaneous Element	Required / Optional	Description	Field in Msg, if applicable
ME1	Required	Component name of Publisher	"COMPONENT" from Bus Header of msg
ME2	Required	How telemetry is packaged (CCSDS frame or packet, or other).	'FORMAT' from the msg content
ME3	Required for CCSDS	Virtual channel ID	"VCID" from CCSDS msg content, or from header portion of CCSDS frame
ME4	Required for CCSDS Packet	AP ID – identifies a particular subsystem on the spacecraft	From header portion of data stream
	N/A for CCSDS Frame		
ME5	Optional	Point on ground system where data was captured	'COLLECTION-POINT' from msg content

Example for Publisher / Sender of Telemetry Messages:

GMSEC.MSSN.SAT1.MSG.TLM.SATSIM.CCSDSFRAME.2.1
GMSEC.MSSN.SAT1.MSG.TLM.TFEP.TDM.1.2

Example for Subscriber / Receiver of Telemetry Messages:

GMSEC.*.SAT1.MSG.TLM.*.CCSDSFRAME.>
GMSEC.*.SAT1.MSG.TLM.*.CCSDSPKT.>

5.3.2 Mnemonic Data Messages

5.3.2.1 Real-Time Mnemonic Value Messages

The Mnemonic Value Messages provide the mechanism for requesting and sending mnemonic data that has been processed from a data stream. A requesting component, such as a trending system, may request a set of mnemonics from a data stream. The request may be for a single set of values or for a continual stream of values based upon a sampling rate or upon change. The responding component, such as a Telemetry and Command system, extracts the set of requested mnemonics from the data stream and sends them to the requesting component. The Mnemonic Value Messages remove the burden from the requesting component of having to process (decommutate) the data and therefore having to know the specifics of the telemetry database. The three specific Archive Mnemonic Value messages are:

- Mnemonic Value Request Message
- Mnemonic Value Response Message
- Mnemonic Value Data Message

5.3.2.1.1 Mnemonic Value Request Message

The Mnemonic Value Request Message is used when an application desires real-time mnemonic data from another application. A common use of the Mnemonic Value Request Message is when a Flight Dynamics application wishes to produce a flight dynamics product, such as ephemeris, orbit, and attitude products. The Flight Dynamics application builds a request of the mnemonic values to be collected and sends this request to the telemetry subsystem. The telemetry subsystem would package the decommutated values and flags for all the requested mnemonics into a Mnemonic Value Data Message and publish it for use.

The Mnemonic Value Request Message is used to subscribe to real-time mnemonics, while the Mnemonic Value Response and Mnemonic Value Data Message are used to publish mnemonics. The Mnemonic Value Request Message is used to request one to n mnemonics either as a single sample, “Oneshot”, or as a request to “Start” publishing the mnemonics continuously. The Mnemonic Value Request Message is also used to “Stop” the publishing of the mnemonics.

Table 5-71. Mnemonic Value Request Message Summary

Sender	A GMSEC compliant application such as a GUI Subsystem, Command Verification process, Expert Subsystem, FDS Process
Senders Intended Usage	Request
Receiver	Any mnemonic processor such as a Telemetry Decommutation process or data archiver
Receivers Intended Usage	Subscribe
What	Spacecraft health and safety data and configuration data values
When	As needed
Quality of Service	Guaranteed

Example:

1. Command verification process requests telemetry value to check if spacecraft command executed as expected
2. Flight Dynamics process requests telemetry values used in the generation of Flight Dynamics products.

Figure 5.3.2.1-1 shows a sequence diagram for the different Mnemonic Value Messages exchanged between the requestor and data provider.

The Mnemonic Value Request Message consists of an Information Bus Header and the Message Body contents. The information bus header identifies the message as a GMSEC Mnemonic Value Request Message. The message contents specify the number of mnemonics being requested, the type of request, the data sampling criteria, the rate the messages should be published, and the mnemonic names.

It is recommended that the data requestor (subscriber) and data provider (publisher) use the Request/Reply functions of the GMSEC API. The Request/Reply function is an easy method of guaranteeing that the mnemonic value request and response messages are exchanged in a point-to-point manner. The data requestor (client) would use the Request function to send the Mnemonic Value Request Message (with a Request-Type of Start or Oneshot), and the data provider (server/publisher) would use the Reply function to send the Mnemonic Value Response Message back to the requestor. If the Request-Type was to "Start", the data provider/publisher/server would publish the Mnemonic Value Data Messages until it was determined they were no longer needed. To conclude the scenario, the data requestor would then again use the Request function to send the Mnemonic Value Request Message to the data provider with a Request-Type of "Stop", and the data provider would use the Reply function to send the final message in the sequence, the Mnemonic Value Response Message.

Please see Section 4.2 GMSEC Messages: Their Characteristics and Interactions for a general discussion on these types of messages.

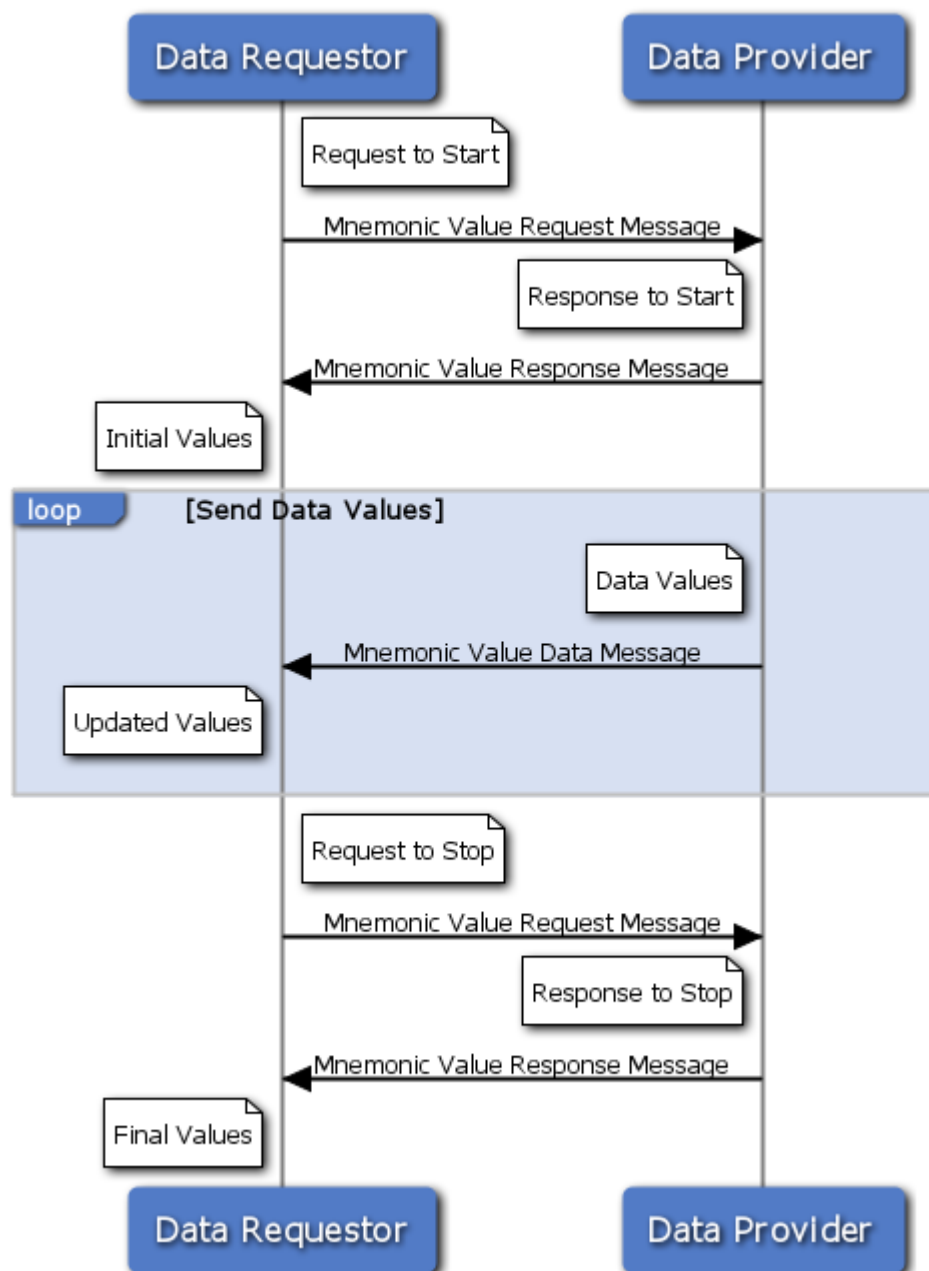


Figure 5.3.2.1-1 Mnemonic Value Message Sequence Diagram

5.3.2.1.1.1 Mnemonic Value Request Message Information Bus Header

The abbreviated table below shows the required Values of the MESSAGE-TYPE and MESSAGE-SUBTYPE fields for this message.

Table 5-72. Mnemonic Value Request Information Bus Header

Field Name	Req/ Opt/API	Value	Type	Notes
Message Information				
HEADER-VERSION	R	2010	F32	Version Number for this message description.
MESSAGE-TYPE	R	REQ	Header string	Message type identifier: REQ, RESP, or MSG
MESSAGE-SUBTYPE	R	MVAL	Header string	Unique message identifier, fixed for GMSEC Standard Messages
More ... Please refer to Table 4-9. GMSEC Information Bus Header for a complete definition.

5.3.2.1.1.2 Mnemonic Value Request Message Contents

Table 5-73. Mnemonic Value Request Message Contents

Field Name	Req/ Opt	Value/Description		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	Unique ID used to distinguish the message
Publishing Information					
REQUEST-TYPE	R	Value	Description	I16	Identifies the type of mnemonic value request message
		1	Oneshot		
		2	Start		
		3	Stop		
PUBLISH-RATE	O	0+		U16	Identifies the rate, in number of seconds, which the Mnemonic Value Data messages are published. Zero means the server should publish the data as fast as possible. Default rate = 5 seconds.
Time Period Selection. Choose Time Window or Duration.					
STRUCTURE: Time Window					
START-TIME	O			Time (absolute or relative)	Requested start time of the mnemonic values.
STOP-TIME	O			Time (absolute or relative)	Requested stop time of the mnemonic values.
DURATION	O	Value	Description	U16	Length of time from “now”, in seconds, for the request to be active, after which the data messages will automatically cease.
		1+			
Downlink Characteristics					
COLLECTION-POINT	O			String	Receiver, device, point, path, etc. where data was received. Used to distinguish data simultaneously received at multiple collection points.
Mnemonic Information					
NUM-OF-MNEMONICS	R	1+		U16	Total Number of mnemonics being requested. Required only for Oneshot and Start.
MNEMONIC.n.NAME	R	“n” starts at “1”		String	Name of the mnemonic. Required only for Oneshot and Start.
MNEMONIC.n.DATA-TYPE	O	Value	Description	I16	Indicates the data type to be returned, either the raw value, or the converted value (Engineering Units or Text converted), or both. Defaults to both.
		1	Raw		
		2	Converted		
		3	Both		
MNEMONIC.n.STATE-ATTRIBUTES	O	Value	Description	I16	Indicates if State Attributes (flags, limits, static flag, and data quality) are to be returned. Defaults to No.
		1	No		
		2	Yes		
MNEMONIC.n.CRITERIA	O	Value	Description	I16	Identification of when data should be provided for the mnemonic. Includes either upon change of data (value, flags or status), or every sample, or at a specified sampling rate. The default Criteria is “Change” only data.
		1	Change (value, flags, status)		
		2	Every Sample		
		3	Sample Rate		
MNEMONIC.n.SAMPLE-RATE	D	1+	milliseconds	U16	If CRITERIA is specified as “Sample Rate”, this field will specify the data sampling rate for the mnemonic.

Oneshot Request: A REQUEST-TYPE of “Oneshot” will result in one set of mnemonic data being returned (the last current value). No further updates will occur. For a “Oneshot” request the following fields or options ARE NOT meaningful:

- PUBLISH-RATE
- DURATION
- MNEMONIC.n.CRITERIA
- MNEMONIC.n.SAMPLE-RATE

Start and Stop Request: The REQUEST-TYPE of “Start” and “Stop” are used for streaming the data in multiple messages. If the MNEMONIC.n.CRITERIA are not specified in the Mnemonic Value Request Message, then the criteria will default to a value of 1 for Change only data. Additionally, if the publish rate is not specified a default publish rate of 5 seconds will be used. A specified publish rate of zero is a request for the data to be published at the fastest rate possible by the data server/provider. The data provider may have a predetermined maximum publish rate, say no faster than 1 message per second, or it may decide to make an on-the-spot calculation of its capabilities based upon its current publishing responsibilities. For example, the data provider may know that it is limited to an output rate of 1 megabyte per second. If it is currently near its maximum output rate and after calculating the additional load of the request it would exceed that rate, the data provider may reject the Mnemonic Value Request with a “Failed Completion” status in the RESPONSE-STATUS field, and an optional status of “Unable to meet demand” in the RETURN-VALUE field.

START-TIME and STOP-TIME: For some real-time mnemonic data requests, the requestor will need to know the time period of the desired data. As an example, a data requestor of a satellite with a 12-hour pass or even a satellite in constant ground contact will need a means to selectively limit the data it receives, rather than take all the data from the pass. To specify data with a time window, the START-TIME and STOP-TIME parameters are to be used.

DURATION: When a data requestor does not specify any START-TIME or STOP-TIME parameters, a potential issue with the Mnemonic Request, Response, and Value Messages is how to halt the endless publication of messages when the requestor fails to request the cessation of Mnemonic Value Data Messages. This could occur by poor design or through equipment failure where the requestor disappears and is no longer alive to request that publication be halted. One option is to have the requestor specify the DURATION of time that the Mnemonics Value Messages should be published (the length of a pass in seconds, for example). At the conclusion of this time period the publisher would automatically cease publication. The advantage to this approach is that no matter what the reason for the requestor failing to request a halt to the publication of messages, they will automatically and eventually stop. Or, the data provider may self-impose a default maximum duration. That is, the provider will

only publish mnemonic values for, say a maximum of 30 minutes, or until a stop request is received.

5.3.2.1.2 Mnemonic Value Response Message

The Mnemonic Value Response Message is used to acknowledge receipt of, and provide status to a Mnemonic Value Request Message. The Response Status in the Mnemonic Value Response Message indicates the success or failure of the component to process the Mnemonic Value Request Message. The ordering of the mnemonics in the Mnemonic Value Response shall be the same as the receiving order specified in the Mnemonic Value Request Message. When an invalid mnemonic is detected in the Mnemonic Value Request Message the following shall apply to the Mnemonic Value Response Message:

- Set the RESPONSE-STATUS field to “5 Invalid Request”
- Set the MNEMONIC.n.STATUS field of the invalid mnemonic(s) to “3 Invalid”
- Set the MNEMONIC.n.STATUS field of all other valid mnemonics to “2 Valid, Nodata”.
- Set the MNEMONIC.n.NUM-OF-SAMPLES to zero for all mnemonics
- The Mnemonic Value Data Messages shall not be published.

The Mnemonic Value Response Message also provides a message time-stamp and the values of the requested mnemonics. For a “Oneshot” Mnemonic Value Request Message, the Mnemonic Value Response Message contains the values of the requested mnemonics, and no further Mnemonic Value Data messages will be published. For a “Start” Mnemonic Value Request Message, the Mnemonic Value Response Message contains the initial values of the requested mnemonics, and subsequent occurrences of the mnemonics will be published via the Mnemonic Value Data Message. For a “Stop” Mnemonic Value Request Message, the Mnemonic Value Response Message contains the last value of the requested mnemonics, and no further Mnemonic Value Data Messages will be published for this group of mnemonics.

The Number of Samples for a mnemonic can be either a zero or a one. In the case of an invalid mnemonic or a valid mnemonic with no data, the Number of Samples shall be set to zero. In the case of a valid mnemonic with a good data sample, the Number of Samples shall be set to one. If there is no data available for a mnemonic, the mnemonic status field will indicate a valid mnemonic with a no data condition and the subsequent data value fields will not be provided for this mnemonic.

The Mnemonic Value Response Message contains the following optional data for the requested mnemonics: raw value, Engineering Units converted value, Units associated with the Engineering Units converted value, text converted value, flags native to the publishing component, red high indicator, red low indicator, yellow high indicator, yellow low indicator, static indicator, and data quality indicator.

Table 5-74. Mnemonic Value Response Message Summary

Sender	A GMSEC compliant application such as a telemetry decommutation process
Senders Intended Usage	Reply
Receiver	GUI Subsystem, Command Verification process, Expert Subsystem, Analysis subsystem
Receivers Intended Usage	Subscribe
What	Acknowledgment and status of Mnemonic Value Request Message
When	Upon receipt of a Mnemonic Value Request Message
Quality of Service	Guaranteed

Example:

1. Command verification process requests telemetry value to check if spacecraft command executed as expected
2. Flight Dynamics process requests telemetry values used in the generation of Flight Dynamics products.

5.3.2.1.2.1 Mnemonic Value Response Message Information Bus Header

The abbreviated table below shows the required Values of the MESSAGE-TYPE and MESSAGE-SUBTYPE fields for this message.

Table 5-75. Mnemonic Value Response Information Bus Header

Field Name	Req/ Opt/API	Value	Type	Notes
Message Information				
HEADER-VERSION	R	2010	F32	Version Number for this message description.
MESSAGE-TYPE	R	RESP	Header string	Message type identifier: REQ, RESP, or MSG
MESSAGE-SUBTYPE	R	MVAL	Header string	Unique message identifier, fixed for GMSEC Standard Messages
More ... Please refer to Table 4-9. GMSEC Information Bus Header for a complete definition.

5.3.2.1.2.2 Mnemonic Value Response Message Contents

Table 5-76. Mnemonic Value Response Message Contents

Field Name	Req/ Opt	Value/Description		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	
STRUCTURE: Response Status					
RESPONSE-STATUS	R	Value	Description	I16	Identifies the status of the Mnemonic Value Request Message that was processed. ("2" is not a valid value and has no meaning.)
		1	Acknowledgement		
		3	Successful Completion		
		4	Failed Completion		
		5	Invalid Request		
TIME-COMPLETED	O			Time	Time application completed processing the request
RETURN-VALUE	O			I32	Return value or status based on the RESPONSE-STATUS. Useful to provide function call status or error code in the case of failed completion
Common Information for All Mnemonics					
NUM-OF-MNEMONICS	R	1+		U16	Total number of mnemonics returned. Should echo the "NUM-OF-MNEMONICS" field in the request message.
Common Information for a Single Mnemonic					
MNEMONIC.1.NAME	R			String	Name of the first Mnemonic
MNEMONIC.1.STATUS	R	Value	Description	I16	Status of the first mnemonic: valid mnemonic or valid mnemonic with no data or invalid mnemonic
		1	Valid		
		2	Valid, Nodata		
		3	Invalid		
MNEMONIC.1.UNITS	O			String	Units associated with the value converted to engineering units for the first mnemonic
MNEMONIC.1.NUM-OF-SAMPLES	R			U16	Number of data samples for the first mnemonic.
Single Data Point Information					
MNEMONIC.1.SAMPLE.1.TIME-STAMP	O			Time	Time stamp for the first data sample of the first mnemonic
MNEMONIC.1.SAMPLE.1.RAW-VALUE	O			I32	Raw value for the first data sample of the first mnemonic
MNEMONIC.1.SAMPLE.1.EU-VALUE	O			F32	Raw value converted to Engineering Units if engineering units conversion is present for the first data sample of the first mnemonic
MNEMONIC.1.SAMPLE.1.TEXT-VALUE	O			String	Raw value converted to a text string if text conversion is present for the first data sample of the first mnemonic
MNEMONIC.1.SAMPLE.1.FLAGS	O			I32	Flags native to the T&C component for the first data sample of the first mnemonic
MNEMONIC.1.SAMPLE.1.LIMIT-ENABLE-DISABLE	O	Value	Description	Boolean	Indicates the limit checking state for the first data sample of the first mnemonic
		0	Disabled		
		1	Enabled		
MNEMONIC.1.SAMPLE.1.RED-HIGH	O	Value	Description	Boolean	Indicates the Red High limit status of the first data sample of the first mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.1.SAMPLE.1.RED-LOW	O	Value	Description	Boolean	Indicates the Red Low limit status of the first data sample of the first mnemonic
		0	In-limits		
		1	Out-of-limits		

MNEMONIC.1.SAMPLE.1.YELLOW-HIGH	O	Value	Description	Boolean	Indicates the Yellow High limit status of the first data sample of the first mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.1.SAMPLE.1.YELLOW-LOW	O	Value	Description	Boolean	Indicates the Yellow Low limit status of the first data sample of the first mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.1.SAMPLE.1.STATIC	O	Value	Description	Boolean	Indicates the static (stale) condition of the first data sample of the first mnemonic
		0	Active		
		1	Static		
MNEMONIC.1.SAMPLE.1.QUALITY	O	Value	Description	Boolean	Indicates the Quality of the first data sample of the first mnemonic
		0	Good quality		
		1	Questionable quality		
: : : : : : :	O	: : : : : : : : : : : : : : : : :		: : : : : : :	: : : : : : : : : : : : : : : : :
Common Information for a Single Mnemonic					
MNEMONIC.n.NAME	R	"n" began with 1		String	Name of the 'n th ' Mnemonic
MNEMONIC.n.STATUS	R	Value	Description	I16	Status of the 'n th ' mnemonic: valid mnemonic or valid mnemonic with no data or invalid mnemonic
		1	Valid		
		2	Valid, Nodata		
		3	Invalid		
MNEMONIC.n.UNITS	O			String	Units associated with the value converted to engineering units for the 'n th ' mnemonic
MNEMONIC.n.NUM-OF-SAMPLES	R			U16	Number of data samples for the 'n th ' mnemonic
Single Data Point Information					
MNEMONIC.n.SAMPLE.n.TIME-STAMP	O			Time	Time stamp for the first data sample of the 'n th ' mnemonic
MNEMONIC.n.SAMPLE.n.RAW-VALUE	O			I32	Raw value for the first data sample of the 'n th ' mnemonic
MNEMONIC.n.SAMPLE.n.EU-VALUE	O			F32	Raw value converted to Engineering Units if engineering units conversion is present for the first data sample of the 'n th ' mnemonic
MNEMONIC.n.SAMPLE.n.TEXT-VALUE	O			String	Raw value converted to a text string if text conversion is present for the first data sample of the 'n th ' mnemonic
MNEMONIC.n.SAMPLE.n.FLAGS	O			I32	Flags native to the T&C component for the first data sample of the 'n th ' mnemonic
MNEMONIC.n.SAMPLE.n.LIMIT-ENABLE-DISABLE	O	Value	Description	Boolean	Indicates the limit checking state for the first data sample of the 'n th ' mnemonic
		0	Disabled		
		1	Enabled		
MNEMONIC.n.SAMPLE.n.RED-HIGH	O	Value	Description	Boolean	Indicates the Red High limit status of the first data sample of the 'n th ' mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.n.SAMPLE.n.RED-LOW	O	Value	Description	Boolean	Indicates the Red Low limit status of the first data sample of the 'n th ' mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.n.SAMPLE.n.YELLOW-HIGH	O	Value	Description	Boolean	Indicates the Yellow High limit status of the first data sample of the 'n th ' mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.n.SAMPLE.n.YELLOW-LOW	O	Value	Description	Boolean	Indicates the Yellow Low limit status of the first data sample of the 'n th ' mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.n.SAMPLE.n.STATIC	O	Value	Description	Boolean	Indicates the static (stale) condition of the first data sample of the 'n th ' mnemonic
		0	Active		
		1	Static		
MNEMONIC.n.SAMPLE.n.QUALITY	O	Value	Description	Boolean	Indicates the Quality of the first data sample of the 'n th ' mnemonic
		0	Good quality		
		1	Questionable quality		

Note: The “I32” data type for the Field Name “MNEMONIC.1.RAW-VALUE.1” may need to be revisited if it is insufficient in length.

The following data attributes are not included in the Mnemonic Value Response or Mnemonic Value Data messages: Delta Limits, Rail Limits, Inverted Limits, and Foreground / Background colors for text values. The Database Attributes messages can provide this information.

5.3.2.1.3 Mnemonic Value Data Message

The Mnemonic Value Data Message provides the telemetry or configuration mnemonic data that was requested in the Mnemonic Value Request Message. The message is generated in response to receiving a Mnemonic Value Request Message to “start” publishing the Mnemonic values for one to n mnemonics and following the generation of a Mnemonic Value Response Message with successful completion status. The Mnemonic Value Data Messages shall not be published if the Mnemonic Value Request Message contained any invalid mnemonics. The Mnemonic Value Data Message will continue to be published at the requested distribution rate until a “Stop” Mnemonic Value Request Message is received, or the DURATION, specified in the Mnemonic Value Request Message has expired. The ordering of the mnemonics in the Mnemonic Value Data Message shall be the same as the receiving order specified in the Mnemonic Value Request Message.

The Mnemonic Value Data Message will contain one to n mnemonics and one to n data samples per mnemonic. The Mnemonic Value Data Message contains the following required data fields: message identifier from the Request Message and the number of mnemonics. The Mnemonic Value Data Message contains the following optional fields for each requested mnemonics: mnemonic name, mnemonic status, and number of samples. The Mnemonic Value Data Message will also contain the following optional data for each data sample of the mnemonic: time of the sample, raw value, Engineering Units converted value, Units associated with the Engineering Units converted value, text converted value, flags native to the publishing component, red high indicator, red low indicator, yellow high indicator, yellow low indicator, static indicator, and data quality indicator. If there is no data available for a mnemonic, the mnemonic status field will indicate a valid mnemonic with a no data condition and the subsequent data value fields will not be provided for this mnemonic.

Table 5-77. Mnemonic Value Data Message Summary

Sender	A GMSEC compliant application such as a telemetry decommutation process
Senders Intended Usage	Publish
Receiver	GUI Subsystem, Command Subsystem, Schedule Execution process, Expert Subsystem
Receivers Intended Usage	Subscribe
What	Spacecraft health and safety data or configuration data values
When	Upon interval requested at a minimum and dependent on data rate and/or replay rate and/or change rate
Quality of Service	Reliable or Guaranteed

Example:

1. Telemetry data to be displayed on a GUI page
2. Telemetry data for an analysis plot

5.3.2.1.3.1 Mnemonic Value Data Message Information Bus Header

The abbreviated table below shows the required Values of the MESSAGE-TYPE and MESSAGE-SUBTYPE fields for this message.

Table 5-78. Mnemonic Value Data Message Information Bus Header

Field Name	Req/ Opt/API	Value	Type	Notes
Message Information				
HEADER-VERSION	R	2010	F32	Version Number for this message description.
MESSAGE-TYPE	R	MSG	Header string	Message type identifier: REQ, RESP, or MSG
MESSAGE-SUBTYPE	R	MVAL	Header string	Unique message identifier, fixed for GMSEC Standard Messages
More ... Please refer to Table 4-9. GMSEC Information Bus Header for a complete definition.

5.3.2.1.3.2 Mnemonic Value Data Message Contents

Table 5-79. Mnemonic Value Data Message Contents

Field Name	Req/ Opt	Value/Description		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
MSG-ID	R			Header String	Echo of the MSG-ID field from the Mnemonic Value Request message.
Indicator for Last Message in Series					
FINAL-MESSAGE	O	Value	Description	Boolean	When true, indicates last message in the series.
		0	No/False		
		1	Yes/True		
Common Information for All Mnemonics					
NUM-OF-MNEMONICS	R			U16	Total number of mnemonics in this message
Common Information for a Single Mnemonic					
MNEMONIC.1.NAME	O			String	Name of the first mnemonic
MNEMONIC.1.STATUS	O	Value	Description	I16	Status of the first mnemonic: valid mnemonic, or valid mnemonic with no data, or invalid mnemonic
		1	Valid		
		2	Valid, No data		
		3	Invalid		
MNEMONIC.1.UNITS	O			String	Units associated with the value converted to engineering units for the first mnemonic
MNEMONIC.1.NUM-OF-SAMPLES	O			U16	Number of data samples for the first mnemonic
First Single Data Point Information for the First Mnemonic					
MNEMONIC.1.SAMPLE.1.TIME-STAMP	O			Time	Time stamp for the first data sample of the first mnemonic
MNEMONIC.1.SAMPLE.1.RAW-VALUE	O			I32	Raw value for the first data sample of the first mnemonic
MNEMONIC.1.SAMPLE.1.EU-VALUE	O			F32	Raw value converted to Engineering Units if engineering units conversion is present for the first data sample of the first mnemonic
MNEMONIC.1.SAMPLE.1.TEXT-VALUE	O			String	Raw value converted to a text string if text conversion is present for the first data sample of the first mnemonic
MNEMONIC.1.SAMPLE.1.FLAGS	O			I32	Flags native to the T&C component for the first data sample of the first mnemonic
MNEMONIC.1.SAMPLE.1.LIMIT-ENABLE-DISABLE	O	Value	Description	Boolean	Indicates the limit checking state for the first data sample of the first mnemonic
		0	Disabled		
		1	Enabled		
MNEMONIC.1.SAMPLE.1.RED-HIGH	O	Value	Description	Boolean	Indicates the Red High limit status for the first data sample of the first mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.1.SAMPLE.1.RED-LOW	O	Value	Description	Boolean	Indicates the Red Low limit status for the first data sample of the first mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.1.SAMPLE.1.YELLOW-HIGH	O	Value	Description	Boolean	Indicates the Yellow High limit status for the first data sample of the first mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.1.SAMPLE.1.YELLOW-LOW	O	Value	Description	Boolean	Indicates the Yellow Low limit status for the first data sample of the first mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.1.SAMPLE.1.STATIC	O	Value	Description	Boolean	Indicates the static (stale) condition for the first data sample of the first
		0	Active		

		1	Static		mnemonic
MNEMONIC.1.SAMPLE. 1.QUALITY	O	Value	Description	Boolean	Indicates the Quality for the first data sample of the first mnemonic
		0	Good quality		
		1	Questionable quality		
: : : : : :	O	: : : : : :	: : : : : :	: : : : : :	: : : : : :
Last Single Data Point Information for the First Mnemonic					
MNEMONIC.1.SAMPLE. n.TIME-STAMP	O			Time	Time stamp for the 'n th ' data sample of the first mnemonic
MNEMONIC.1.SAMPLE. n.RAW-VALUE	O			I32	Raw value for the 'n th ' data sample of the first mnemonic
MNEMONIC.1.SAMPLE. n.EU-VALUE	O			F32	Raw value converted to Engineering Units if engineering units conversion is present for the 'n th ' data sample of the first mnemonic
MNEMONIC.1.SAMPLE. n.TEXT-VALUE	O			String	Raw value converted to a text string if text conversion is present for the 'n th ' data sample of the first mnemonic
MNEMONIC.1.SAMPLE. n.FLAGS	O			I32	Flags native to the T&C component for the 'n th ' data sample of the first mnemonic
MNEMONIC.1.SAMPLE. n.LIMIT-ENABLE- DISABLE	O	Value	Description	Boolean	Indicates the limit checking state for the 'n th ' data sample of the first mnemonic
		0	Disabled		
		1	Enabled		
MNEMONIC.1.SAMPLE. n.RED-HIGH	O	Value	Description	Boolean	Indicates the Red High limit status for the 'n th ' data sample of the first mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.1.SAMPLE. n.RED-LOW	O	Value	Description	Boolean	Indicates the Red Low limit status for the 'n th ' data sample of the first mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.1.SAMPLE. n.YELLOW-HIGH	O	Value	Description	Boolean	Indicates the Yellow High limit status for the 'n th ' data sample of the first mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.1.SAMPLE. n.YELLOW-LOW	O	Value	Description	Boolean	Indicates the Yellow Low limit status for the 'n th ' data sample of the first mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.1.SAMPLE. n.STATIC	O	Value	Description	Boolean	Indicates the static (stale) condition for the 'n th ' data sample of the first mnemonic
		0	Active		
		1	Static		
MNEMONIC.1.SAMPLE. n.QUALITY	O	Value	Description	Boolean	Indicates the Quality for the 'n th ' data sample of the first mnemonic
		0	Good quality		
		1	Questionable quality		
: : : : : :	O	: : : : : :	: : : : : :	: : : : : :	: : : : : :
Common Information for the Last Mnemonic					
MNEMONIC.n.NAME	O			String	Name of the 'n th ' mnemonic
MNEMONIC.n.STATUS	O	Value	Description	I16	Status of the 'n th ' mnemonic: valid mnemonic, or valid mnemonic with nodata, or invalid mnemonic
		1	Valid		
		2	Valid, Nodata		
		3	Invalid		
MNEMONIC.n.UNITS	O			String	Units associated with the raw value converted to engineering units for the 'n th ' mnemonic
MNEMONIC.n.NUM-OF-SAMPLES	O			U16	Number of data samples for the 'n th ' mnemonic
First Single Data Point Information for the Last Mnemonic					
MNEMONIC.n.SAMPLE. 1.TIME-STAMP	O			Time	Time stamp for the first data sample of the 'n th ' mnemonic
MNEMONIC.n.SAMPLE. 1.RAW-VALUE	O			I32	Raw value for the first data sample of the 'n th ' mnemonic
MNEMONIC.n.SAMPLE. 1.EU-VALUE	O			F32	Raw value converted to Engineering Units if engineering units conversion

				is present for the first data sample of the 'n th ' mnemonic	
MNEMONIC.n.SAMPLE.1.TEXT-VALUE	O		String	Raw value converted to a text string if text conversion is present for the first data sample of the 'n th ' mnemonic	
MNEMONIC.n.SAMPLE.1.FLAGS	O		I32	Flags native to the T&C component for the first data sample of the 'n th ' mnemonic	
MNEMONIC.n.SAMPLE.1.LIMIT-ENABLE-DISABLE	O	Value	Description	Boolean	Indicates the limit checking state for the first data sample of the 'n th ' mnemonic
		0	Disabled		
		1	Enabled		
MNEMONIC.n.SAMPLE.1.RED-HIGH	O	Value	Description	Boolean	Indicates the Red High limit status of the value for the first data sample of the 'n th ' mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.n.SAMPLE.1.RED-LOW	O	Value	Description	Boolean	Indicates the Red Low limit status of the telemetry value for the first data sample of the 'n th ' mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.n.SAMPLE.1.YELLOW-HIGH	O	Value	Description	Boolean	Indicates the Yellow High limit status of the telemetry value for the first data sample of the 'n th ' mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.n.SAMPLE.1.YELLOW-LOW	O	Value	Description	Boolean	Indicates the Yellow Low limit status of the telemetry value for the first data sample of the 'n th ' mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.n.SAMPLE.1.STATIC	O	Value	Description	Boolean	Indicates the static (stale) condition of the telemetry value for the first data sample of the 'n th ' mnemonic
		0	Active		
		1	Static		
MNEMONIC.n.SAMPLE.1.QUALITY	O	Value	Description	Boolean	Indicates the Quality of the telemetry values for the first data sample of the 'n th ' mnemonic
		0	Good quality		
		1	Questionable quality		
: : : : : :	O	: : : : : :	: : : : : :	: : : : : :	
Last Single Data Point Information for the Last Mnemonic					
MNEMONIC.n.SAMPLE.n.TIME-STAMP	O		Time		Time stamp for the 'n th ' data sample of the 'n th ' mnemonic
MNEMONIC.n.SAMPLE.n.RAW-VALUE	O		I32		Raw value for the 'n th ' data sample of the 'n th ' mnemonic
MNEMONIC.n.SAMPLE.n.EU-VALUE	O		F32		Raw value converted to Engineering Units if engineering units conversion is present for the 'n th ' data sample of the 'n th ' mnemonic
MNEMONIC.n.SAMPLE.n.TEXT-VALUE	O		String		Raw value converted to a text string if text conversion is present for the 'n th ' data sample of the 'n th ' mnemonic
MNEMONIC.n.SAMPLE.n.FLAGS	O		I32		Flags native to the T&C component for the 'n th ' data sample of the 'n th ' mnemonic
MNEMONIC.n.SAMPLE.n.LIMIT-ENABLE-DISABLE	O	Value	Description	Boolean	Indicates the limit checking state for the 'n th ' data sample of the 'n th ' mnemonic
		0	Disabled		
		1	Enabled		
MNEMONIC.n.SAMPLE.n.RED-HIGH	O	Value	Description	Boolean	Indicates the Red High limit status for the 'n th ' data sample of the 'n th ' mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.n.SAMPLE.n.RED-LOW	O	Value	Description	Boolean	Indicates the Red Low limit status for the 'n th ' data sample of the 'n th ' mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.n.SAMPLE.n.YELLOW-HIGH	O	Value	Description	Boolean	Indicates the Yellow High limit status for the 'n th ' data sample of the 'n th ' mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.n.SAMPLE.n.YELLOW-LOW	O	Value	Description	Boolean	Indicates the Yellow Low limit status for the 'n th ' data sample of the 'n th ' mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.n.SAMPLE.n.STATIC	O	Value	Description	Boolean	Indicates the static (stale) condition for the 'n th ' data sample of the 'n th ' mnemonic
		0	Active		
		1	Static		

MNEMONIC.n.SAMPLE. n.QUALITY	O	Value	Description	Boolean	Indicates the Quality for the 'n th ' data sample of the 'n th ' mnemonic
		0	Good quality		
		1	Questionable quality		

5.3.2.1.4 Mnemonic Value Data Message, Request, and Response Message Subjects

Table 5-80. Mnemonic Value Request Message Subject Naming

	Subject Standard	Mission Elements		Message Elements		Miscellaneous Elements	
Subject Element	Specification	MISSION	SAT	TYP	SUBTYP	ME1	ME2
Subject Content	GMSEC	[mission]	[sat]	REQ	MVAL	[Component: TLM3, TLM2, ...]	
Example for Publisher / Sender	GMSEC	MSSN	SAT1	REQ	MVAL	TLM3	
Example for Publisher / Sender	GMSEC	MSSN	SAT1	REQ	MVAL	TLM2	
Example for Subscriber / Receiver	GMSEC	*	SAT1	REQ	MVAL	TLM3	

Table 5-81. Properties of the *Miscellaneous Elements* for the Mnemonic Value Request Message

<i>Miscellaneous Element</i>	Required / Optional	Description	Field in Msg, if applicable
ME1	Required	Component name of Responder	NA
ME2	Not used		

Examples:

Two components, APP5 and TLM3 interact with the Mnemonic Value Request Message.

TLM3 subscribes to receive the Mnemonic Value Request Message.

GMSEC.*.REQ.MVAL.TLM3

GMSEC.MSSN.SAT1.REQ.*.TLM3 (TLM3 will receive any REQ msg)

APP5 (Data Requestor/Subscriber/Client) sends a request to TLM3, the (Data Provider/Publisher/Server).

GMSEC.MSSN.SAT1.REQ.MVAL.TLM3

Table 5-82. Mnemonic Value Response Message Subject Naming

	Subject Standard	Mission Elements		Message Elements		<i>Miscellaneous Elements</i>		
Subject Element	Specification	MISSION	SAT	TYP	SUBTYP	ME1	ME2	ME3
Subject Content	GMSEC	[mission]	[sat]	RESP	MVAL	[Component: TLM3, TLM2, ...]	[Status]	[]
Example for Publisher / Sender	GMSEC	MSSN	SAT1	RESP	MVAL	TLM3	1	
Example for Publisher / Sender	GMSEC	MSSN	SAT1	RESP	MVAL	APP5	1	
Example for Subscriber / Receiver	GMSEC	*	SAT1	RESP	MVAL	TLM3	*	

Table 5-83. Properties of the *Miscellaneous Elements* for the Mnemonic Value Response Message

<i>Miscellaneous Element</i>	Required / Optional	Description	Field Origination in Msg, if applicable
ME1	Required	Component name of Requestor	Echo of "COMPONENT" in header of Request msg
ME2	Required	Status type supplied by Responder	"RESPONSE-STATUS" from content of Response message

Examples:

Two components FD (the Data Requestor/Subscriber/Client) and TLM3 (the Data Provider/Publisher/Server) interact with the Mnemonic Value Response Message.

FD subscribe subject to receive the Mnemonic Value Response Message.

GMSEC.MSSN.*.RESP.MVAL.FD.> or
GMSEC.*.*.RESP.MVAL.FD.>

TLM3 sends a response message to FD.

GMSEC.MSSN.SAT1.RESP.MVAL.FD.2 or
GMSEC.MSSN.SAT1.RESP.MVAL.FD.4

Table 5-84. Mnemonic Value Data Message Subject Naming

	Subject Standard	Mission Elements		Message Elements		<i>Miscellaneous Elements</i>		
Subject Element	Specification	MISSION	SAT	TYP	SUBTYP	ME1	ME2	ME3
Subject Content	GMSEC	[mission]	[sat]	MSG	MVAL	[Component: TLM3, TLM2, ...]		
Example for Publisher / Sender	GMSEC	MSSN	SAT1	MSG	MVAL	TLM3		
Example for Publisher / Sender	GMSEC	MSSN	SAT1	MSG	MVAL	TLM2		
Example for Subscriber / Receiver	GMSEC	*	SAT1	MSG	MVAL	TLM3		

Table 5-85. Properties of the *Miscellaneous Elements* for the Mnemonic Value Data Message

<i>Miscellaneous Element</i>	Required / Optional	Description	Field in Msg, if applicable
ME1	Required	Component name of Publisher	Echo of "COMPONENT" in header of Request msg
ME2	Not Used		
ME3	Not used		

Examples:

After the successful exchange of the Mnemonic Value Request and Response Messages, the Mnemonic Value Data Message is published.

The Requestor/Subscriber/Client subscribes to receive the intended Mnemonic Value Data Messages.

GMSEC.MSSN.SAT1.MSG.MVAL.*

The Data Provider/Publisher/Server sends out the Mnemonic Value Data Message with the following subject:

GMSEC.MSSN.SAT1.MSG.MVAL.TLM3

GMSEC.MSSN.SAT1.MSG.MVAL.TLM2

5.3.2.2 Archive Mnemonic Value Messages

The Archive Mnemonic Value Messages provide a mechanism for requesting and delivering mnemonic data that has been stored in an archive. A requesting component, such as a trending system, may request a set of mnemonics from the data archive. The request is generally for a set of values, over an interval of time, and at a desired data sampling rate. The responding component, such as an Archive system, extracts the set of requested mnemonics from the data archive and provides them to the requesting component via a variety of available delivery methods. The Archive Mnemonic Value Messages remove the burden from the requesting component of having to process (decommutate) the data from the archive and therefore having to know the specifics of the telemetry database and the structure of the data archive. The Archive Mnemonic Value Messages also remove the burden from the requesting component from hardware and software associated with the creation, management, and maintenance of a data archive. The three specific Archive Mnemonic Value messages are:

- Archive Mnemonic Value Request Message
- Archive Mnemonic Value Response Message
- Archive Mnemonic Value Data Message

5.3.2.2.1 Archive Mnemonic Value Request Message

The Archive Mnemonic Value Request Message is used when an application desires mnemonic data from previously recorded data that has been stored in the Telemetry Archive. A common use of the Archive Mnemonic Value Request Message is when an analysis component wishes to produce a trending product, such as Battery charge/discharge, or a propellant graph. The Analysis component builds a request of the mnemonics to be retrieved from the telemetry archive and sends this request to an Archive Management component. The Archive Management Component may be a stand-alone component or it may be part of another subsystem, such as a T&C subsystem.

The Archive Management component will package the decommutated values and their associated attributes (if desired) for all the requested mnemonics, over the time frame requested. A number of extraction options and delivery methods are available for choosing. These include:

- Time interval
 - The requestor is required to specify the time span of the desired data
- Data selection
 - The requested data can be for raw, converted, or both types of data
 - Attributes (flags, limits, static, quality) can be included or not

- Data sampling can specify all, upon change, or at a periodic sample rate
- Data delivery method is either by
 - One single response message, or
 - As a stream of messages similar to a real-time mnemonic data values
 - If this delivery method is selected, the requestor can also select the speed of the data delivery, either as kilobits per second or as a ratio of the real-time rate.
- Actual data or by reference
 - The data can be within the response message or a URI can reference the location of a data file
- If the requestor has asked for a data file, the attributes of the file can be further specified.
 - Using product specifications, and
 - File specifications

In summary, a variety of data selection criteria and data delivery mechanisms are available to customize and best match the needs of the requestor. Many of the fields in the Archive Mnemonic Value Request Message are optional and dependent on other fields, but they also have specified default values so that only a minimal number of fields need be actually specified to extract and deliver the data. Of course, the more specific and customized the data request, the more fields that will need to be specified.

Table 5-86. Archive Mnemonic Value Request Message Summary

Sender	A GMSEC compliant application such as an Analysis and Assessment component
Senders Intended Usage	Request
Receiver	Any Archive mnemonic processor such as an Archive Management component or a T&C component
Receivers Intended Usage	Subscribe
What	Spacecraft health and safety data, ground configuration data, or any data stored with a mnemonic name
When	As needed
Quality of Service	Guaranteed

Example:

1. An Analysis process requests telemetry values to create a graph of a spacecraft instrument's performance
2. A Flight Dynamics process requests telemetry values used in the generation of Flight Dynamics products.

The Archive Mnemonic Value Request Message consists of an information bus header and the message contents. The information bus header identifies the message as a GMSEC Archive Mnemonic Value Request Message. The message contents specify the time range, the data sampling criteria, and the mnemonic names. The message contents also provides for the specification of the delivery method of the data.

5.3.2.2.1.1 Archive Mnemonic Value Request Message Information Bus Header

The abbreviated table below shows the required Values of the MESSAGE-TYPE and MESSAGE-SUBTYPE fields for this message.

Table 5-87. Archive Mnemonic Value Request Information Bus Header

Field Name	Req/ Opt/API	Value	Type	Notes
Message Information				
HEADER-VERSION	R	2010	F32	Version Number for this message description.
MESSAGE-TYPE	R	REQ	Header string	Message type identifier: REQ, RESP, or MSG
MESSAGE-SUBTYPE	R	AMVAL	Header string	Unique message identifier, fixed for GMSEC Standard Messages
More ... Please refer to Table 4-9. GMSEC Information Bus Header for a complete definition.

5.3.2.2.1.2 Archive Mnemonic Value Request Message Contents

Table 5-88. Archive Mnemonic Value Request Message Contents

Field Name	Req/ Opt	Value/Description		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	Unique ID used to distinguish the message
Data Delimiters					
STRUCTURE: Time Window					
START-TIME	R			Time (absolute or relative)	Requested start time of the mnemonic values to be retrieved from the telemetry archive.
STOP-TIME	O			Time (absolute or relative)	Requested stop time of the mnemonic values to be retrieved from the telemetry archive. Defaults to the end of the telemetry archive
PDB-VERSION	O			String	Project Data Base version to be used by the responder when processing the archived data. Defaults to the PDB version used when the data was archived.
Downlink Characteristics					
COLLECTION-POINT	O			String	Receiver, device, point, path, etc. where data was received. Used to distinguish data simultaneously received at multiple collection points.
Product Distribution Information					
RESPOND-VIA-MSG	R	Value	Description	String	Indicates the message to use to deliver the mnemonic data. MSG will be a stream of messages; RESP will be a single response message.
		“MSG.AMV AL”	AMVAL Message		
		“RESP.AM VAL”	AMVAL Response Message		
STRUCTURE: Product Distribution Options					
DELIVER-VIA-REFERENCE	D	0	No / False	Boolean	This parameter is used only if “RESP.AMVAL” is selected above. Indicates if the data will be referenced by a URI in the single response message. Defaults to No.
		1	Yes / True		
DELIVER-VIA-INCLUDE	D	0	No / False	Boolean	This parameter is used only if “RESP.AMVAL” is selected above. Indicates if the data is to be included in the single response message. Defaults to Yes.
		1	Yes / True		
Replay Speed by Playback Ratio or Data Rate – Choose One					
PLAYBACK-RATIO	D	> 0 and < 1 is slower than real-time rate = 1 is equal to the real-time rate > 1 is faster than real-time rate		F32	If “MSG.AMVAL” is selected above, specifies the speed of data delivery as a ratio of playback rate to real-time rate.
DATA-RATE	D	> 0		I16	If “MSG.AMVAL” is selected above, specifies the speed of data delivery in Kilobits per second
STRUCTURE: Output Product Category Identification (used when RESP.AMVAL has been specified above)					
PROD-NAME	D			String	The “PROD-” fields are optionally used when the “RESP.AMVAL” has been specified above. Name of the product being requested.

PROD-DESCRIPTION	O			String	Description of the product in text or xml
PROD-TYPE	D	Value	Description	String	Product type and subtype being requested. (See Table 4-29. Product Categories)
PROD-SUBTYPE	D	AAA	Archive and Assessment		
		DATA		String	
NUM-OF-PROD-SUBTYPES	D			I16	Number of further delineations / categories beyond the product subtype. Also, used as msg subject elements ME5, ME6, etc. in Product Message.
PROD-SUBTYPE.n.NAME	D			String	First subcategory of the product subtype. (Subject elements ME5, ME6, etc. of the Product Message)
STRUCTURE: Output File Attributes					
URI	D			String	Location where the requesting component is asking for the product file(s) to be stored. Could be a web address, directory or folder specification
NAME-PATTERN	D			String	Describes the name of the output file
DESCRIPTION	D			String	Description of the file in text or xml
FORMAT	D			String	Describes the file format
VERSION	D			String	Identifies the version of the file
SIZE	D	Kilobytes		U32	Maximum size of the file acceptable to the requester.
Mnemonic Information					
NUM-OF-MNEMONICS	R			U16	Total Number of mnemonics being requested
MNEMONIC.n.NAME	R	"n" starts at "1"		String	Name of the mnemonic
MNEMONIC.n.DATA-TYPE	O	Value	Description	I16	Indicates the data type to be returned, either the raw value, or the converted value (Engineering Units or Text converted), or both. Defaults to both.
		1	Raw		
		2	Converted		
		3	Both		
MNEMONIC.n.STATE-ATTRIBUTES	O	Value	Description	I16	Indicates if the State Attributes (flags, limits, static flag, and data quality) of the mnemonic are to be returned. Defaults to No.
		1	No		
		2	Yes		
MNEMONIC.n.CRITERIA	O	Value	Description	I16	Identification of how data should be sampled for the mnemonic. Includes either upon change of data (value, flags or status), or every sample, or at a specified sampling rate. Defaults to "Change" only data.
		1	Change (value, flags, status)		
		2	Every Sample		
		3	Sample Rate		
MNEMONIC.n.SAMPLE-RATE	D	1+	milliseconds	U16	If CRITERIA is specified as "Sample Rate", this field will specify the data sampling rate for the mnemonic.

For an explanation on how the START-TIME and STOP-TIME could operate, see Table 5-19. Examples of Start and Stop Times.

The archived mnemonic data can be delivered in a

1. **Stream of messages**, akin to the stream of real-time mnemonic data value messages, or a
2. **Single response** message.

1. STREAM OF MESSAGES DATA DELIVERY

The advantage of delivering the archived mnemonic data as a stream of messages is that the processing can be similar if not identical to the procedure used with the real-time Mnemonic Value Data Messages. To use this delivery mechanism, specify the following:

- Set the field RESPOND-VIA-MSG to the value “MSG.AMVAL”
- Optionally, specify the speed of data delivery with either of the fields “PLAYBACK-RATIO” or “DATA-RATE”
- Identify the number and names of the mnemonics along with their extraction criteria

2. SINGLE RESPONSE MESSAGE DATA DELIVERY

The advantage of delivering the archived mnemonic data in a single response message is that the data is entirely contained within one location and can be processed in bulk. When using this data delivery mechanism, the requestor has a few options on how and where the data is to be delivered. The requested data will be delivered in a file. The requestor can ask for the data file:

1. Within the single response message,
2. By reference, using a URI within the single response message, or
3. Both.

To accomplish the desired result, each of these options is explained below.

First, for all the above options using a single response message:

- Set the field RESPOND-VIA-MSG to the value “RESP.AMVAL”

Second, choose one of the three following data delivery options:

1. **INCLUDE WITHIN MESSAGE:** To include only the data file within the single response message (with no URI reference), the requestor should do the following:
 - Nothing! The “DELIVER-VIA-“ fields will default to include the file within the single response message with no URI reference. No other field under the Product Distribution Options section is required to be specified.
2. **BY REFERENCE:** To only have the data file specified by reference and NOT be included in the single response message, the requestor should do the following:
 - Set the field DELIVER-VIA-REFERENCE to the value “Yes/True”
 - Set the field DELIVER-VIA-INCLUDE to the value “No/False”
3. **BOTH:** To have both the data file included in the single response message AND specified as a reference the requestor should do the following:
 - Set the field DELIVER-VIA-REFERENCE to the value “Yes/True”
 - The field DELIVER-VIA-INCLUDE will default to the value “Yes/True”

Third, for all the options above, the requestor must do the following:

- Identify the number and names of the mnemonics along with their extraction criteria
- Optionally, if known or applicable, the requestor can specify the file type and file attributes with the optional dependent fields under the Output Product Category and Output File Attributes sections.

Other features of note for this request message are:

- When specifying the mnemonics, if the MNEMONIC.n.CRITERIA is not specified in the Archive Mnemonic Value Request Message, then the criteria will default to a value of 1 for Change only data.
- If a URI has been specified in the request, it is assumed that the DELIVER-VIA-REFERENCE field was set to “Yes/True”. If the URI was specified, the resulting archive mnemonic value product will be “pushed” to the location specified by the URI.
- If the URI has not been specified in the request, but the DELIVER-VIA-REFERENCE field was set to “Yes/True”, then the resulting archive mnemonic value product will be copied to a URI location designated by the provider of the data. This URI location must also be included in the Archive Mnemonic Value Response Message. The requestor of the data file can “pull” the file using the provided URI.
- If the component servicing the Archive Mnemonic Value Request Message supports several formats or versions of a product, then the requesting component can specify the format or version of the resulting product in the FORMAT and VERSION fields. If the FORMAT and/or VERSION fields are not specified, then the component servicing the Archive Mnemonic Value Request message shall default to the latest format or version of its product.

5.3.2.2.2 Archive Mnemonic Value Response Message

An Archive Provider in response to an Archive Mnemonic Value Request Message sends an Archive Mnemonic Value Response Message. The job of the Archive Mnemonic Value Response Message is to provide acknowledgment of the Archive Mnemonic Value Request Message, the overall status of the completed action, the specific status of each requested mnemonic, and optionally, if requested, the resulting data file and its associated attributes. A series of Archive Mnemonic Value Response Messages may be required. In this case, an initial acknowledgement response message is issued, followed by interim or interactive “working” response type messages to let the requesting application know that the request is still being processed, and finally a completion response type message. If an audit trail or operator notification is required, the requesting application is responsible for generating a Log Message indicating the result of the Archive Mnemonic Value Request Message. Please see Section 4.2 GMSEC Messages: Their Characteristics and Interactions for a general discussion on these types of messages.

Table 5-89. Archive Mnemonic Value Response Message Summary

Sender	A GMSEC compliant application that has access to a telemetry archive such as a T&C component
Senders Intended Usage	Reply
Receiver	Assessment and Analysis component
Receivers Intended Usage	Subscribe
What	Acknowledgment and status of Archive Mnemonic Value Request Message
When	Upon receipt of an Archive Mnemonic Value Request Message and/or completion of the Request
Quality of Service	Guaranteed

Example:

1. An Archive Manager responds to a component in the Assessment and Analysis subsystem that requested battery telemetry values to check the spacecraft battery rate of charge/discharge
2. The Archive Manager responds to a component in the Flight Dynamics subsystem that requested telemetry values to be used in the generation of a Flight Dynamics product.

5.3.2.2.2.1 Archive Mnemonic Value Response Message Information Bus Header

The abbreviated table below shows the required Values of the MESSAGE-TYPE and MESSAGE-SUBTYPE fields for this message.

Table 5-90. Archive Mnemonic Value Response Information Bus Header

Field Name	Req/ Opt/API	Value	Type	Notes
Message Information				
HEADER-VERSION	R	2010	F32	Version Number for this message description.
MESSAGE-TYPE	R	RESP	Header string	Message type identifier: REQ, RESP, or MSG
MESSAGE-SUBTYPE	R	AMVAL	Header string	Unique message identifier, fixed for GMSEC Standard Messages
More ... Please refer to Table 4-9. GMSEC Information Bus Header for a complete definition.

5.3.2.2.2 Archive Mnemonic Value Response Message Contents

Table 5-91. Archive Mnemonic Value Response Message Contents

Field Name	Req/ Opt	Value/Description		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	.
STRUCTURE: Response Status					
RESPONSE-STATUS	R	Value	Description	I16	Identifies the status of the Archive Mnemonic Value Request Message that was processed.
		1	Acknowledgement		
		2	Working / Keep Alive		
		3	Successful Completion		
		4	Failed Completion		
		5	Invalid Request		
	6	Final Message			
TIME-COMPLETED	O			Time	Time application completed processing the request
RETURN-VALUE	O			I32	Return value or status based on the RESPONSE-STATUS. Useful to provide function call status or error code in the case of failed completion
STRUCTURE: Product Category Identification					
PROD-NAME	O			String	Name of the product
PROD-DESCRIPTION	O			String	Description of the product in text or xml
PROD-TYPE	O	Value	Description	String	Product type and subtype being requested. (See Table 4-29. Product Categories)
		AAA	Archive and Assessment		
PROD-SUBTYPE	O	DATA		String	
NUM-OF-PROD-SUBTYPES	O			U16	Number of further delineations / categories beyond the product subtype. Also, used as msg subject elements ME5, ME6, etc. in Product Message.
PROD-SUBTYPE.n.NAME	O			String	First subcategory of the product subtype. (Subject elements ME5, ME6, etc. of the Product Message)
STRUCTURE: Output Product Information					
URI	O			String	URI specifying the location where the (single) output file product is stored
NAME-PATTERN	O			String	Describes the name of the output file
DESCRIPTION	O			String	Description of the file in text or xml
FORMAT	O			String	Describes the file format
VERSION	O			String	Identifies the version of the file
SIZE	O	Kilobytes		U32	Actual size of the file
STRUCTURE: Data File					
DATA	O			Binary (blob)	The file content
Mnemonic Information					
NUM-OF-MNEMONICS	D			U16	Total number of mnemonics returned
MNEMONIC.n.NAME	D	“n” starts at “1”		String	Name of the ‘n th ’ Mnemonic
MNEMONIC.n.STATUS	D	Value	Description	I16	Status of the ‘n th ’ mnemonic: valid mnemonic or valid mnemonic with no data or invalid mnemonic
		1	Valid		
		2	Valid, Nodata		
		3	Invalid		

The RESPONSE-STATUS field in the Archive Mnemonic Value Response Message indicates the success or failure of the component to process the Archive Mnemonic Value Request Message. The values returned in the RESPONSE-STATUS field will indicate if the request message was received,

valid, invalid, able to be successfully and completely processed, or if the processing failed. The ordering of the mnemonics in the Archive Mnemonic Value Response shall be the same as the receiving order specified in the Archive Mnemonic Value Request Message. If any of the requested mnemonics in the Archive Mnemonic Value Request Message are invalid, the following are to occur:

- Set the RESPONSE-STATUS field to “5” or “Invalid Request”
- Set the status field of the invalid mnemonic(s) to “3” or “Invalid” (MNEMONIC.n.STATUS)
- Set the status field of all other valid mnemonics to “2” or “Valid, Nodata”.
- The Archive Mnemonic Value product shall not be generated

The MNEMONIC.n.STATUS field provides the status of each requested mnemonic. These dependent fields indicate:

- Valid – mnemonic was validated and data was located that met the criteria in the Archive Mnemonic Request Message
- Valid, nodata – mnemonic was validated, but no data met the criteria in the corresponding request message
- Invalid – mnemonic was not found in the database or list of mnemonics

The following table indicates when the dependent fields in the response message are required.

Table 5-92. Relationship between RESPONSE-STATUS and Dependent Fields

Value	Description	Dependent Fields Required?
1	Acknowledgement	N
2	Working/Keep Alive	N
3	Successful Completion	Y
4	Failed Completion	N
5	Invalid Request	Y
6	Final Message	N

If a request is invalid (RESPONSE-STATUS field = “Invalid Request”) it could be because one or more of the requested mnemonics was invalid. For this return status, the responder should provide all the requested mnemonics and the status of each.

The requestor has the option of specifying the URI where the responder should place the product file. If the URI is not specified, the responder will place the product file in its own designated location and return that location in the URI field of the response message. If the requestor specifies the URI, the responder will place the product file in that location, if possible, and return that same URI from the request message in the response message along with the corresponding RESPONSE-STATUS and RETURN-VALUE values. If the responder is unable

to place the product file in the specified URI location, the responder will place the file in an alternate URI location and return the URI in the response message along with the corresponding RESPONSE-STATUS and RETURN-VALUE values. It is possible that the responder cannot access (write to) the URI specified by the requestor, and neither can the requestor access (read from) the alternate URI chosen by the responder in which case they will need to work out access and protection issues. When the RESPONSE-STATUS is successful, the RETURN-VALUE can have the following status indicators:

- 1 – product file was placed in requestor's designated location
- 2 – product file was placed in provider's designated location
- 3 – product file was generated in format other than that requested

The following table shows the relationship between the URI, RESPONSE-STATUS, and the RETURN-VALUE.

Table 5-93. Interpretation of the RESPONSE-STATUS and RETURN-VALUE Fields

User Specified the URI	RESPONSE-STATUS	RETURN-VALUE	URI	Action
N	Successful	2	Product was generated and placed in URI chosen by responder	Requestor should retrieve file at responder's URI location
Y	Successful	1	Product was generated and placed in URI specified by requestor	Requestor should retrieve file at the specified URI
Y	Successful	2	Product was generated but placed in alternate URI chosen by responder	Requestor should retrieve file at responder's URI location
NA	Successful	3	NA	Requestor should retrieve file at the specified URI.

The requestor also has the option of specifying the format and version of the product file. If the FORMAT and VERSION are not specified, the responder will use the latest file format and version to build the product. If the requestor specified the FORMAT and VERSION, the responder will generate the product in the desired format and version, if possible. If the responder is unable to generate the product in the format and version requested, the responder will generate the product in the latest format and version along with the corresponding RESPONSE-STATUS and RETURN-VALUE.

5.3.2.2.3 Archive Mnemonic Value Data Message

The Archive Mnemonic Value Data Message provides the telemetry or configuration mnemonic data that was requested in the Archive Mnemonic Value Request Message. The messages are generated in response to receiving an Archive Mnemonic Value Request Message to publish the requested Mnemonic values in a stream of messages. (In this case, the Archive Mnemonic Value Request Message specified the delivery mechanism to be a stream of messages - similar to the real-time Mnemonic Value Data Messages.) Figure 5.3.2.2.3-1 shows a sequence diagram for the different Archive Mnemonic Value Messages and how the message protocol between the data requestor and data provider would occur.

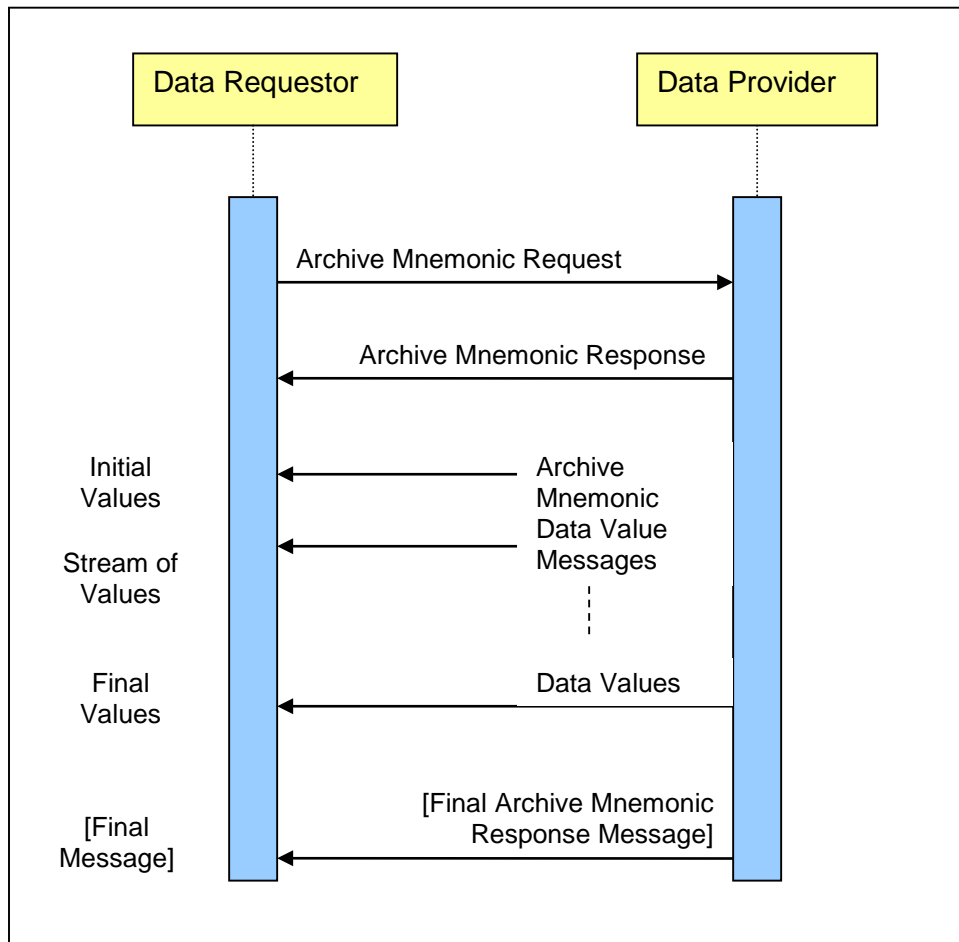


Figure 5.3.2.2.3-1 Archive Mnemonic Value Message Sequence Diagram

The previous diagram shows an initial exchange of the

- Archive Mnemonic Value Request Message and
- Archive Mnemonic Value Response Message

This is followed by a stream of

- Archive Mnemonic Value Data Messages

A final message is optional and can be sent, if desired. It is the

- Archive Mnemonic Value Response Message

The stream of Archive Mnemonic Value Data Messages follows the successful exchange of the Archive Mnemonic Value Request and Response Messages. The Archive Mnemonic Value Data Messages shall not be published if the Archive Mnemonic Value Request Message contained any invalid mnemonics. The Archive Mnemonic Value Data Messages will continue to be published at the specified rate until all requested data has been published. The ordering of the mnemonics in the Archive Mnemonic Value Data Message shall be the same as the order specified in the Archive Mnemonic Value Request Message.

The Archive Mnemonic Value Data Message will contain one to many mnemonics and one to many data samples per mnemonic. Besides the required CONTENT-VERSION field, the Archive Mnemonic Value Data Message also requires an additional field: the number of mnemonics (NUM-OF-MNEMONICS). All other fields are considered optional, mostly due to what was specified for inclusion in the Archive Mnemonic Value Request Message. The Archive Mnemonic Value Data Message contains the following optional fields for each requested mnemonic: mnemonic name, mnemonic status, and number of samples. The Mnemonic Value Data Message will also contain the following optional data for each data sample of the mnemonic: time of the sample, raw value, Engineering Units converted value, Units associated with the Engineering Units converted value, text converted value, flags native to the publishing component, red high indicator, red low indicator, yellow high indicator, yellow low indicator, static indicator, and data quality indicator. If there is no data available for a mnemonic, the MNEMONIC.n.STATUS field will indicate a "Valid, Nodata" condition exists and the subsequent associated data value fields will not be provided for this mnemonic.

Table 5-94. Archive Mnemonic Value Data Message Summary

Sender	A GMSEC compliant application that has access to a telemetry archive such as a T&C component
Senders Intended Usage	Publish
Receiver	GUI Subsystem, Command Subsystem, Schedule Execution process, Expert Subsystem, Assessment & Analysis
Receivers Intended Usage	Subscribe
What	Spacecraft health and safety data, configuration data values, any mnemonic data value
When	After sending successful Archive Mnemonic Value Response Message. Then, at specified interval or requested rate.
Quality of Service	Reliable or Guaranteed

Example:

1. Telemetry data to be displayed on a GUI page
2. Telemetry data for an analysis plot

5.3.2.2.3.1 Archive Mnemonic Value Data Message Information Bus Header

The abbreviated table below shows the required Values of the MESSAGE-TYPE and MESSAGE-SUBTYPE fields for this message.

Table 5-95. Archive Mnemonic Value Data Message Information Bus Header

Field Name	Req/ Opt/API	Value	Type	Notes
Message Information				
HEADER-VERSION	R	2010	F32	Version Number for this message description.
MESSAGE-TYPE	R	MSG	Header string	Message type identifier: REQ, RESP, or MSG
MESSAGE-SUBTYPE	R	AMVAL	Header string	Unique message identifier, fixed for GMSEC Standard Messages
More ... Please refer to Table 4-9. GMSEC Information Bus Header for a complete definition.

5.3.2.2.3.2 Archive Mnemonic Value Data Message Contents

Table 5-96. Archive Mnemonic Value Data Message Contents

Field Name	Req/ Opt	Value/Description		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	
Indicator for Last Message in Series					
FINAL-MESSAGE	O	Value	Description	Boolean	When true, indicates the last message in the stream.
		0	No/False		
		1	Yes/True		
Common Information for All Mnemonics					
NUM-OF-MNEMONICS	R			U16	Total number of mnemonics in this message
Common Information for a Single Mnemonic					
MNEMONIC.1.NAME	O			String	Name of the first mnemonic
MNEMONIC.1.STATUS	O	Value	Description	I16	Status of the first mnemonic: valid mnemonic, or valid mnemonic with no data, or invalid mnemonic
		1	Valid		
		2	Valid, No data		
		3	Invalid		
MNEMONIC.1.UNITS	O			String	Units associated with the value converted to engineering units for the first mnemonic
MNEMONIC.1.NUM-OF-SAMPLES	O			U16	Number of data samples for the first mnemonic
First Single Data Point Information for the First Mnemonic					
MNEMONIC.1.SAMPLE.1.TIME-STAMP	O			Time	Time stamp for the first data sample of the first mnemonic
MNEMONIC.1.SAMPLE.1.RAW-VALUE	O			I32	Raw value for the first data sample of the first mnemonic
MNEMONIC.1.SAMPLE.1.EU-VALUE	O			F32	Raw value converted to Engineering Units if engineering units conversion is present for the first data sample of the first mnemonic
MNEMONIC.1.SAMPLE.1.TEXT-VALUE	O			String	Raw value converted to a text string if text conversion is present for the first data sample of the first mnemonic
MNEMONIC.1.SAMPLE.1.FLAGS	O			I32	Flags native to the T&C component for the first data sample of the first mnemonic
MNEMONIC.1.SAMPLE.1.LIMIT-ENABLE-DISABLE	O	Value	Description	Boolean	Indicates the limit checking state for the first data sample of the first mnemonic
		0	Disabled		
		1	Enabled		
MNEMONIC.1.SAMPLE.1.RED-HIGH	O	Value	Description	Boolean	Indicates the Red High limit status for the first data sample of the first mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.1.SAMPLE.1.RED-LOW	O	Value	Description	Boolean	Indicates the Red Low limit status for the first data sample of the first mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.1.SAMPLE.1.YELLOW-HIGH	O	Value	Description	Boolean	Indicates the Yellow High limit status for the first data sample of the first mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.1.SAMPLE.1.YELLOW-LOW	O	Value	Description	Boolean	Indicates the Yellow Low limit status for the first data sample of the first mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.1.SAMPLE.1.STATIC	O	Value	Description	Boolean	Indicates the static (stale) condition for the first data sample of the first mnemonic
		0	Active		
		1	Static		

MNEMONIC.1.SAMPLE. 1.QUALITY	O	Value	Description	Boolean	Indicates the Quality for the first data sample of the first mnemonic
		0	Good quality		
		1	Questionable quality		
.....	O
Last Single Data Point Information for the First Mnemonic					
MNEMONIC.1.SAMPLE. n.TIME-STAMP	O			Time	Time stamp for the 'n th ' data sample of the first mnemonic
MNEMONIC.1.SAMPLE. n.RAW-VALUE	O			I32	Raw value for the 'n th ' data sample of the first mnemonic
MNEMONIC.1.SAMPLE. n.EU-VALUE	O			F32	Raw value converted to Engineering Units if engineering units conversion is present for the 'n th ' data sample of the first mnemonic
MNEMONIC.1.SAMPLE. n.TEXT-VALUE	O			String	Raw value converted to a text string if text conversion is present for the 'n th ' data sample of the first mnemonic
MNEMONIC.1.SAMPLE. n.FLAGS	O			I32	Flags native to the T&C component for the 'n th ' data sample of the first mnemonic
MNEMONIC.1.SAMPLE. n.LIMIT-ENABLE- DISABLE	O	Value	Description	Boolean	Indicates the limit checking state for the 'n th ' data sample of the first mnemonic
		0	Disabled		
MNEMONIC.1.SAMPLE. n.RED-HIGH	O	Value	Description	Boolean	Indicates the Red High limit status for the 'n th ' data sample of the first mnemonic
		0	In-limits		
MNEMONIC.1.SAMPLE. n.RED-LOW	O	Value	Description	Boolean	Indicates the Red Low limit status for the 'n th ' data sample of the first mnemonic
		0	In-limits		
MNEMONIC.1.SAMPLE. n.YELLOW-HIGH	O	Value	Description	Boolean	Indicates the Yellow High limit status for the 'n th ' data sample of the first mnemonic
		0	In-limits		
MNEMONIC.1.SAMPLE. n.YELLOW-LOW	O	Value	Description	Boolean	Indicates the Yellow Low limit status for the 'n th ' data sample of the first mnemonic
		0	In-limits		
MNEMONIC.1.SAMPLE. n.STATIC	O	Value	Description	Boolean	Indicates the static (stale) condition for the 'n th ' data sample of the first mnemonic
		0	Active		
MNEMONIC.1.SAMPLE. n.QUALITY	O	Value	Description	Boolean	Indicates the Quality for the 'n th ' data sample of the first mnemonic
		0	Good quality		
.....	O
Common Information for the Last Mnemonic					
MNEMONIC.n.NAME	O			String	Name of the 'n th ' mnemonic
MNEMONIC.n.STATUS	O	Value	Description	I16	Status of the 'n th ' mnemonic: valid mnemonic, or valid mnemonic with nodata, or invalid mnemonic
		1	Valid		
		2	Valid, Nodata		
MNEMONIC.n.UNITS	O			String	Units associated with the raw value converted to engineering units for the 'n th ' mnemonic
MNEMONIC.n.NUM-OF-SAMPLES	O			U16	Number of data samples for the 'n th ' mnemonic
First Single Data Point Information for the Last Mnemonic					
MNEMONIC.n.SAMPLE. 1.TIME-STAMP	O			Time	Time stamp for the first data sample of the 'n th ' mnemonic
MNEMONIC.n.SAMPLE. 1.RAW-VALUE	O			I32	Raw value for the first data sample of the 'n th ' mnemonic
MNEMONIC.n.SAMPLE. 1.EU-VALUE	O			F32	Raw value converted to Engineering Units if engineering units conversion is present for the first data sample of

MNEMONIC.n.SAMPLE.1.TEXT-VALUE	O			String	the 'n th ' mnemonic Raw value converted to a text string if text conversion is present for the first data sample of the 'n th ' mnemonic
MNEMONIC.n.SAMPLE.1.FLAGS	O			I32	Flags native to the T&C component for the first data sample of the 'n th ' mnemonic
MNEMONIC.n.SAMPLE.1.LIMIT-ENABLE-DISABLE	O	Value	Description	Boolean	Indicates the limit checking state for the first data sample of the 'n th ' mnemonic
		0	Disabled		
		1	Enabled		
MNEMONIC.n.SAMPLE.1.RED-HIGH	O	Value	Description	Boolean	Indicates the Red High limit status of the value for the first data sample of the 'n th ' mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.n.SAMPLE.1.RED-LOW	O	Value	Description	Boolean	Indicates the Red Low limit status of the telemetry value for the first data sample of the 'n th ' mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.n.SAMPLE.1.YELLOW-HIGH	O	Value	Description	Boolean	Indicates the Yellow High limit status of the telemetry value for the first data sample of the 'n th ' mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.n.SAMPLE.1.YELLOW-LOW	O	Value	Description	Boolean	Indicates the Yellow Low limit status of the telemetry value for the first data sample of the 'n th ' mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.n.SAMPLE.1.STATIC	O	Value	Description	Boolean	Indicates the static (stale) condition of the telemetry value for the first data sample of the 'n th ' mnemonic
		0	Active		
		1	Static		
MNEMONIC.n.SAMPLE.1.QUALITY	O	Value	Description	Boolean	Indicates the Quality of the telemetry values for the first data sample of the 'n th ' mnemonic
		0	Good quality		
		1	Questionable quality		
: : : : : : : :	O	: : : : : : : : : : : : : : : : : :		: : : : : : : : : :	: : : : : : : : : : : : : : : : : :
Last Single Data Point Information for the Last Mnemonic					
MNEMONIC.n.SAMPLE.n.TIME-STAMP	O			Time	Time stamp for the 'n th ' data sample of the 'n th ' mnemonic
MNEMONIC.n.SAMPLE.n.RAW-VALUE	O			I32	Raw value for the 'n th ' data sample of the 'n th ' mnemonic
MNEMONIC.n.SAMPLE.n.EU-VALUE	O			F32	Raw value converted to Engineering Units if engineering units conversion is present for the 'n th ' data sample of the 'n th ' mnemonic
MNEMONIC.n.SAMPLE.n.TEXT-VALUE	O			String	Raw value converted to a text string if text conversion is present for the 'n th ' data sample of the 'n th ' mnemonic
MNEMONIC.n.SAMPLE.n.FLAGS	O			I32	Flags native to the T&C component for the 'n th ' data sample of the 'n th ' mnemonic
MNEMONIC.n.SAMPLE.n.LIMIT-ENABLE-DISABLE	O	Value	Description	Boolean	Indicates the limit checking state for the 'n th ' data sample of the 'n th ' mnemonic
		0	Disabled		
		1	Enabled		
MNEMONIC.n.SAMPLE.n.RED-HIGH	O	Value	Description	Boolean	Indicates the Red High limit status for the 'n th ' data sample of the 'n th ' mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.n.SAMPLE.n.RED-LOW	O	Value	Description	Boolean	Indicates the Red Low limit status for the 'n th ' data sample of the 'n th ' mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.n.SAMPLE.n.YELLOW-HIGH	O	Value	Description	Boolean	Indicates the Yellow High limit status for the 'n th ' data sample of the 'n th ' mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.n.SAMPLE.n.YELLOW-LOW	O	Value	Description	Boolean	Indicates the Yellow Low limit status for the 'n th ' data sample of the 'n th ' mnemonic
		0	In-limits		
		1	Out-of-limits		
MNEMONIC.n.SAMPLE.n.STATIC	O	Value	Description	Boolean	Indicates the static (stale) condition for the 'n th ' data sample of the 'n th ' mnemonic
		0	Active		
		1	Static		
MNEMONIC.n.SAMPLE.n.QUALITY	O	Value	Description	Boolean	Indicates the Quality for the 'n th ' data sample of the 'n th ' mnemonic

n.QUALITY		0	Good quality		sample of the 'n th ' mnemonic
		1	Questionable quality		

5.3.2.2.4 Archive Mnemonic Value Message Subjects

Table 5-97. Archive Mnemonic Value Request Message Subject Naming

	Subject Standard	Mission Elements		Message Elements		Miscellaneous Elements	
Subject Element	Specification	MISSION	SAT	TYP	SUBTYP	ME1	ME2
Subject Content	GMSEC	[mission]	[sat]	REQ	AMVAL	[Component: TLM3, TLM2, ...]	
Example for Publisher / Sender	GMSEC	MSSN	SAT1	REQ	AMVAL	TLM3	
Example for Publisher / Sender	GMSEC	MSSN	SAT1	REQ	AMVAL	TLM2	
Example for Subscriber / Receiver	GMSEC	MSSN	SAT1	REQ	AMVAL	>	

Table 5-98. Properties of the *Miscellaneous Elements* for the Archive Mnemonic Value Request Message

Miscellaneous Element	Required / Optional	Description	Field in Msg, if applicable
ME1	Required	Component name of Responder	NA
ME2	Not used		

Examples:

Two components, FD (Data Requestor/Subscriber/Client) and TLM3 (Data Provider/Publisher/Server), interact with the Archive Mnemonic Value Request Message.

FD sends the Archive Mnemonic Value Request Message to TLM3.

GMSEC.MSSN.SAT1.REQ.AMVAL.TLM3

TLM3 subscribes to receive the Archive Mnemonic Value Request Message from FD.

GMSEC.MSSN.SAT1.REQ.AMVAL.TLM3 or
GMSEC.*.REQ.AMVAL.>

Table 5-99. Archive Mnemonic Value Response Message Subject Naming

	Subject Standard	Mission Elements		Message Elements		<i>Miscellaneous Elements</i>		
Subject Element	Specification	MISSION	SAT	TYP	SUBTYP	ME1	ME2	ME3
Subject Content	GMSEC	[mission]	[sat]	RESP	AMVAL	[Component: FD, TLM3, TLM2, ...]	[Response Status: 1-ack,...4-failed]	[Response Status: 1-ack,...4-failed]
Example for Publisher / Sender	GMSEC	MSSN	SAT1	REQ	AMVAL	FD	1	
Example for Publisher / Sender	GMSEC	MSSN	SAT1	RESP	AMVAL	FD	1	
Example for Subscriber / Receiver	GMSEC	MSSN	SAT1	RESP	AMVAL	*	>	>

Table 5-100. Properties of the *Miscellaneous Elements* for the Archive Mnemonic Value Response Message

<i>Miscellaneous Element</i>	Required / Optional	Description	Field Origination in Msg, if applicable
ME1	Required	Component name of Requestor	Echo of "COMPONENT" in header of Request msg
ME2	Required	Status type supplied by Responder	"RESPONSE-STATUS" from content of Response message

Examples:

Two components, TLM2 (Data Provider/Publisher/Server) and FD (Data Requestor/Subscriber/Client), interact with the Archive Mnemonic Value Response Message.

TLM2 message subject to send the Archive Mnemonic Value Response Message to FD (and another response message to FD):

GMSEC.MSSN.SAT1.RESP.AMVAL.FD.2
GMSEC.MSSN.SAT1.RESP.AMVAL.FD.3

FD subscribes to receive the Archive Mnemonic Value Response Message from TLM2.

GMSEC.MSSN.SAT1.RESP.AMVAL.FD.>

Table 5-101. Archive Mnemonic Value Data Message Subject Naming

	Subject Standard	Mission Elements		Message Elements		<i>Miscellaneous Elements</i>		
Subject Element	Specification	MISSION	SAT	TYP	SUBTYP	ME1	ME2	ME3
Subject Content	GMSEC	[mission]	[sat]	MSG	AMVAL	[Component: TLM3, TLM2, ...]		
Example for Publisher / Sender	GMSEC	MSSN	SAT1	MSG	AMVAL	TLM3		
Example for Publisher / Sender	GMSEC	MSSN	SAT1	MSG	AMVAL	TLM2		
Example for Subscriber / Receiver	GMSEC	*	SAT1	MSG	AMVAL	TLM3		

Table 5-102. Properties of the *Miscellaneous Elements* for the Mnemonic Value Data Message

<i>Miscellaneous Element</i>	Required / Optional	Description	Field in Msg, if applicable
ME1	Required	Component name of Publisher	Echo of "COMPONENT" in header of Request msg
ME2	Not used		

Examples:

After the successful exchange of the Archive Mnemonic Value Request and Response Messages, the Archive Mnemonic Value Data Messages are published.

Two different Data Provider/Publisher/Servers send out the Archive Mnemonic Value Data Messages with the following subjects:

GMSEC.MSSN.SAT1.MSG.AMVAL.TLM3.

GMSEC.MSSN.SAT1.MSG.AMVAL.TLM2.

The Requestor/Subscriber/Client subscribes to receive the intended Archive Mnemonic Value Data Messages.

GMSEC.MSSN.SAT1.MSG.AMVAL.*.

5.3.2.3 Database Attributes Messages

The Database Attributes Messages provides the mechanism for requesting and sending project database information about a mnemonic. A requesting component, such as a trending system, may request the attributes of a set of mnemonics from the project database. The requested Database Attributes may be for the following:

- A complete list of mnemonics defined in the database
- The database defined limit set values for a list of mnemonics
- The database defined text conversion for a list of mnemonics
- The database defined short or long textual description for a list of mnemonics
- The database defined calibration curve for a list of mnemonics

The responding component, such as a T&C system, retrieves the requested mnemonics attributes from the database and provides them to the requesting component. The Database Attributes Messages removes the burden, from the requesting component, of having to access the project database and therefore having to know the specifics of the telemetry database and its structure. The Database Attributes Message also removes the burden, from the requesting component, for hardware and software associated with the storage and configuration management of the project database. The provider of the database attributes does not necessarily have to be a database or database manager, but some component with the attribute information.

5.3.2.3.1 Database Attributes Request Message

The Database Attributes Request Message is used when an application desires attributes or information about the project database. A common use of the Database Attributes Request Message is when an analysis component wishes to produce a trending product, such as Battery charge/discharge, or a propellant graph. The analysis component may need the limit ranges of the mnemonics for annotation on the graph. The analysis component builds a request of the mnemonics to be retrieved from the project database and sends this request to a provider of the database information. The database provider may be a stand-alone component or it may be part of a T&C component. The Database component will respond with the Database Attributes for the requested mnemonics in a Database Attributes Response Message.

Table 5-103. Database Attributes Request Message Summary

Sender	A GMSEC compliant application such as an Analysis and Assessment component
Senders Intended Usage	Request
Receiver	Any Database processor such as a Database component or a T&C component
Receivers Intended Usage	Subscribe
What	A mnemonic's Database Attributes
When	As needed
Quality of Service	Reliable

Example:

1. An Analysis process requests the limit ranges defined in the project database of a set of telemetry mnemonics for annotation on an analysis graph.

The Database Attributes Request Message consists of an information bus header and the message contents. The information bus header identifies the message as a GMSEC Database Attributes Request Message. The message contents specify the number of mnemonics being requested and the mnemonic names. The message contents also provides for the specification of the type of Database Attributes being requested.

5.3.2.3.1.1 Database Attributes Request Message Information Bus Header

The abbreviated table below shows the required Values of the MESSAGE-TYPE and MESSAGE-SUBTYPE fields for this message.

Table 5-104. Database Attributes Request Information Bus Header

Field Name	Req/ Opt/API	Value	Type	Notes
Message Information				
HEADER-VERSION	R	2010	F32	Version Number for this message description.
MESSAGE-TYPE	R	REQ	Header string	Message type identifier: REQ, RESP, or MSG
MESSAGE-SUBTYPE	R	DB	Header string	Unique message identifier, fixed for GMSEC Standard Messages
More ... Please refer to Table 4-9. GMSEC Information Bus Header for a complete definition.

5.3.2.3.1.2 Database Attributes Request Message Contents

Table 5-105. Database Attributes Request Message Contents

Field Name	Req/ Opt	Value/Description		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	Unique ID used to distinguish the message
General Mnemonic Information					
ATTRIBUTE-TYPE	R	Value	Description	I16	Indicates the type of Database Attributes to be returned.
		1	Limit sets		
		2	Text conversion		
		3	Calibration Curve		
		4	Short Description		
		5	Long Description		
	6	List of All Mnemonics			
PDB-VERSION	O			String	Project Data Base version to be used by the responder if multiple project databases are supported. Defaults to the most recent PDB version.
STRUCTURE: Output Product Category Identification					
PROD-NAME	O			String	Name of the product
PROD-DESCRIPTION	O			String	Description of the product in text or xml
PROD-TYPE	O	Value	Description	String	Product type and subtype being requested. (See Table 4-29. Product Categories)
		TAC	Telemetry and Command		
PROD-SUBTYPE	O	DB		String	
NUM-OF-PROD-SUBTYPES	O	1+		U16	Number of further delineations / categories beyond the product subtype. Also, used as msg subject elements <i>ME5</i> , <i>ME6</i> , etc. in Product Message.
PROD-SUBTYPE.n.NAME	O			String	First subcategory of the product subtype. (Subject elements <i>ME5</i> , <i>ME6</i> , etc. of the Product Message)
STRUCTURE: Output File Attributes					
URI	D			String	Location where the requesting component is asking for the product file(s) to be stored. Could be a web address, directory or folder specification
NAME-PATTERN	O			String	Describes the name of the output file
DESCRIPTION	O			String	Description of the file in text or xml
FORMAT	O			String	Describes the file format
VERSION	O			String	Identifies the version of the file
SIZE	O	KB		U32	Maximum size of the file acceptable to the requester. Size specified in KB.
Mnemonic Information					
NUM-OF-MNEMONICS	D			U16	Total Number of mnemonics being requested
MNEMONIC.1.NAME	D			String	Name of the first mnemonic
: : : : : :	D	: : : : : : : : : : : :	: : : : : : : : : : :	:	: : : : : : : : : : : : : : : : : :
MNEMONIC.n.NAME	D			String	Name of the "n th " mnemonic

Table 5-106. Relationship Between ATTRIBUTE-TYPE and Dependent Fields

Value	Description	Dependent Fields That Are Required
1	Limit sets	Mnemonic Information fields
2	Text conversion	Mnemonic Information fields
3	Calibration Curve	Mnemonic Information fields
4	Short Description	Mnemonic Information fields
5	Long Description	Mnemonic Information fields
6	List of All Mnemonics	URI field

5.3.2.3.2 Database Attributes Response Message

A Database Provider in response to a Database Attributes Request Message sends a Database Attributes Response Message. The purpose of the Response message is to provide acknowledgment of the Database Attributes Request Message, the status of the action completed, information about the output product, and the requested attributes from the project database. The ordering of the mnemonics in the Database Attributes Response Message shall be the same as the receiving order specified in the Database Attributes Request Message. If operator or audit trail notification is required, the requesting application is responsible for generating a Log Message indicating the result of the Database Attributes Request Message.

Table 5-107. Database Attributes Response Message Summary

Sender	A GMSEC compliant application that has access to the project database such as a T&C component
Senders Intended Usage	Reply
Receiver	Requestor of the database attributes
Receivers Intended Usage	Subscribe
What	Acknowledgment, status and project Database Attributes
When	Upon receipt of a Database Attributes Request Message
Quality of Service	Guaranteed

Example:

1. An analysis process requests the limit ranges defined in the project database of a set of telemetry mnemonics for annotation on an analysis graph.

The RESPONSE-STATUS in the Database Attributes Response Message indicates the success or failure of the component to process the Database Attributes Request Message. The MNEMONIC.n.STATUS field indicates if the mnemonic was valid and defined in the project database, or if the mnemonic was invalid. If any of the requested mnemonics in the Database Attributes Request Message are invalid, the following are to occur:

- Set the RESPONSE-STATUS field to “5 - Invalid Request”
- Set the status field of the invalid mnemonic(s) to “3 - Invalid” (MNEMONIC.n.STATUS)
- Set the status field of all other valid mnemonics to “1 - Valid”
- Include values for the following dependent fields:
 - NUM-OF-MNEMONICS
 - MNEMONIC.n.NAME
 - MNEMONIC.n.STATUS
- The Database Attributes shall not be returned if the Request Message contains any invalid mnemonics

The presence of the dependent fields in the Response Message depends upon the value of the RESPONSE-STATUS field. The dependent fields provide the status of each mnemonic that was requested. The table below indicates when the dependent fields are required.

Table 5-108. Relationship between RESPONSE-STATUS and Dependent Fields

Value	Description	Dependent Fields Required?
1	Acknowledgement	N
2	Working/Keep Alive	N
3	Successful Completion	Y
4	Failed Completion	N
5	Invalid Request	Partial, see above

5.3.2.3.2.1 Database Attributes Response Message Information Bus Header

The abbreviated table below shows the required Values of the MESSAGE-TYPE and MESSAGE-SUBTYPE fields for this message.

Table 5-109. Database Attributes Response Information Bus Header

Field Name	Req/ Opt/API	Value	Type	Notes
Message Information				
HEADER-VERSION	R	2010	F32	Version Number for this message description.
MESSAGE-TYPE	R	RESP	Header string	Message type identifier: REQ, RESP, or MSG
MESSAGE-SUBTYPE	R	DB	Header string	Unique message identifier, fixed for GMSEC Standard Messages
More ... Please refer to Table 4-9. GMSEC Information Bus Header for a complete definition.

5.3.2.3.2.2 Database Attributes Response Message Contents

Table 5-110. Database Attributes Response Message Contents for Limit Sets

Field Name	Req/ Opt	Value/Description		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	Unique ID used to distinguish the message
STRUCTURE: Response Status					
RESPONSE-STATUS	R	Value	Description	I16	Identifies the status of the Database Attributes Request Message that was processed.
		1	Acknowledgement		
		2	Working / Keep Alive		
		3	Successful Completion		
		4	Failed Completion		
	5	Invalid Request			
TIME-COMPLETED	O			Time	Time application completed processing the request
RETURN-VALUE	O			I32	Return value or status based on the RESPONSE-STATUS. Useful to provide function call status or error code in the case of failed completion
Attribute / Mnemonic Information					
ATTRIBUTE-TYPE	R	Value	Description	I16	Indicates the type of Database Attributes for the first mnemonic to be returned.
		1	Limit sets		
NUM-OF-MNEMONICS	D			U16	Total number of mnemonics returned
MNEMONIC.n.NAME	D	"n" starts at "1"		String	Name of the Mnemonic
MNEMONIC.n.STATUS	D	Value	Description	I16	Status of the mnemonic: valid mnemonic or valid mnemonic with no data or invalid mnemonic
		1	Valid		
		2	Valid, Nodata		
		3	Invalid		
MNEMONIC.n.LIMIT-TYPE	D	Value	Description	I16	Specifies the type of limits defined for this mnemonic
		1	Range Limits		
		2	Delta Limits		
		3	Both		
MNEMONIC.n.DELTA-LIMIT	D			I16	Delta limit set for Mnemonic n
MNEMONIC.n.MNEMONIC-SWITCH	D			String	Dependent mnemonic used to determine which limit set to use in the case of multiple limit sets.
MNEMONIC.n.NUM-OF-RANGE-SETS	D			I16	Indicates the number of range limit sets defined for this mnemonic
MNEMONIC.n.RANGE-SET.n.LIMIT-CONDITION	D	Value	Description	I16	Indicates if the limit set m for mnemonic n contains Inclusive or Exclusive limit ranges.
		1	Inclusive Limit Set		
		2	Exclusive Limit Set		
MNEMONIC.n.RANGE-SET.n.RH	D			I32	Limit set m Red high limit for Mnemonic n
MNEMONIC.n.RANGE-SET.n.RL	D			I32	Limit set m Red low limit for Mnemonic n
MNEMONIC.n.RANGE-SET.n.YH	D			I32	Limit set m Yellow high limit for Mnemonic n
MNEMONIC.n.RANGE-SET.n.YL	D			I32	Limit set m Yellow low limit for Mnemonic n

Table 5-111. Database Attributes Response Message Contents for Text Conversion

Field Name	Req/ Opt	Value/Description		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	Unique ID used to distinguish the message
STRUCTURE: Response Status					
RESPONSE-STATUS	R	Value	Description	I16	Identifies the status of the Database Attributes Request Message that was processed.
		1	Acknowledgement		
		2	Working / Keep Alive		
		3	Successful Completion		
		4	Failed Completion		
5	Invalid Request				
TIME-COMPLETED	O			Time	Time application completed processing the request
RETURN-VALUE	O			I32	Return value or status based on the RESPONSE-STATUS. Useful to provide function call status or error code in the case of failed completion
Attribute / Mnemonic Information					
ATTRIBUTE-TYPE	R	Value	Description	I16	Indicates the type of Database Attributes for the first mnemonic to be returned.
		2	Text conversion		
NUM-OF-MNEMONICS	D			U16	Total number of mnemonics returned
MNEMONIC.n.NAME	D	"n" starts at 1		String	Name of the Mnemonic
MNEMONIC.n.STATUS	D	Value	Description	I16	Status of the mnemonic: valid mnemonic or valid mnemonic with no data or invalid mnemonic
		1	Valid		
		2	Valid, Nodata		
		3	Invalid		
MNEMONIC.n.NUM-OF-TEXT-SETS	D			I16	Indicates the number of Text Conversion sets defined for this mnemonic
MNEMONIC.n.TEXT-SET.n.TEXT	D			String	Text Conversion Set n for Mnemonic n
MNEMONIC.n.TEXT-SET.n.LOW-RANGE	D			I16	Low range for text conversion set m for mnemonic n
MNEMONIC.n.TEXT-SET.n.HIGH-RANGE	D			I16	High range for text conversion set m for mnemonic n

Table 5-112. Database Attributes Response Message Contents for Calibration Curve

Field Name	Req/ Opt	Value/Description		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	Unique ID used to distinguish the message
STRUCTURE: Response Status					
RESPONSE-STATUS	R	Value	Description	I16	Identifies the status of the Database Attributes Request Message that was processed.
		1	Acknowledgement		
		2	Working / Keep Alive		
		3	Successful Completion		
		4	Failed Completion		
		5	Invalid Request		
TIME-COMPLETED	O			Time	Time application completed processing the request
RETURN-VALUE	O			I32	Return value or status based on the RESPONSE-STATUS. Useful to provide function call status or error code in the case of failed completion
Attribute / Mnemonic Information					
ATTRIBUTE-TYPE	R	Value	Description	I16	Indicates the type of Database Attributes for the first mnemonic to be returned.
		3	Calibration Curve		
NUM-OF-MNEMONICS	D			U16	Total number of mnemonics returned
MNEMONIC.n.NAME	D	"n" starts at "1"		String	Name of the Mnemonic
MNEMONIC.n.STATUS	D	Value	Description	I16	Status of the mnemonic: valid mnemonic or valid mnemonic with no data or invalid mnemonic
		1	Valid		
		2	Valid, Nodata		
		3	Invalid		
MNEMONIC.n.CAL-TYPE	D	Value	Description	I16	Indicates the type of Calibration Curve defined for this mnemonic
		1	Polynomial		
		2	Other		
MNEMONIC.n.CONSTANT	D			F32	Constant for Calibration Curve for mnemonic
MNEMONIC.n.NUM-OF-COEFFS	D			I16	Degree of Polynomial for mnemonic
MNEMONIC.n.COEFF.n.VALUE	D			F32	M th order Polynomial coefficient for MNEMONIC n

Table 5-113. Database Attributes Response Message Contents for Short Description

Field Name	Req/ Opt	Value/Description		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	Unique ID used to distinguish the message
STRUCTURE: Response Status					
RESPONSE-STATUS	R	Value	Description	I16	Identifies the status of the Database Attributes Request Message that was processed.
		1	Acknowledgement		
		2	Working / Keep Alive		
		3	Successful Completion		
		4	Failed Completion		
	5	Invalid Request			
TIME-COMPLETED	O			Time	Time application completed processing the request
RETURN-VALUE	O			I32	Return value or status based on the RESPONSE-STATUS. Useful to provide function call status or error code in the case of failed completion
Attribute / Mnemonic Information					
ATTRIBUTE-TYPE	R	Value	Description	I16	Indicates the type of Database Attributes for the first mnemonic to be returned.
		4	Short Description		
NUM-OF-MNEMONICS	D			U16	Total number of mnemonics returned
MNEMONIC.n.NAME	D	“n” starts at 1		String	Name of the Mnemonic
MNEMONIC.n.STATUS	D	Value	Description	I16	Status of the mnemonic: valid mnemonic or valid mnemonic with no data or invalid mnemonic
		1	Valid		
		2	Valid, Nodata		
		3	Invalid		
MNEMONIC.n.SHORT-DESCRIPTION	D			String	Short Description for mnemonic n

Table 5-114. Database Attributes Response Message Contents for Long Description

Field Name	Req/ Opt	Value/Description		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	Unique ID used to distinguish the message
STRUCTURE: Response Status					
RESPONSE-STATUS	R	Value	Description	I16	Identifies the status of the Database Attributes Request Message that was processed.
		1	Acknowledgement		
		2	Working / Keep Alive		
		3	Successful Completion		
		4	Failed Completion		
TIME-COMPLETED	O			Time	Time application completed processing the request
RETURN-VALUE	O			I32	Return value or status based on the RESPONSE-STATUS. Useful to provide function call status or error code in the case of failed completion
Attribute / Mnemonic Information					
ATTRIBUTE-TYPE	R	Value	Description	I16	Indicates the type of Database Attributes for the first mnemonic to be returned.
		5	Long Description		
NUM-OF-MNEMONICS	D			U16	Total number of mnemonics returned
MNEMONIC.n.NAME	D	"n" starts at "1"		String	Name of the Mnemonic
MNEMONIC.n.STATUS	D	Value	Description	I16	Status of the mnemonic: valid mnemonic or valid mnemonic with no data or invalid mnemonic
		1	Valid		
		2	Valid, Nodata		
		3	Invalid		
MNEMONIC.n.LONG-DESCRIPTION	D			String	Long Description for mnemonic

Table 5-115. Database Attributes Response Message Contents for List of All Mnemonics

Field Name	Req/ Opt	Value/Description		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	Unique ID used to distinguish the message
STRUCTURE: Response Status					
RESPONSE-STATUS	R	Value	Description	I16	Identifies the status of the Database Attributes Request Message that was processed.
		1	Acknowledgement		
		2	Working / Keep Alive		
		3	Successful Completion		
		4	Failed Completion		
		5	Invalid Request		
TIME-COMPLETED	O			Time	Time application completed processing the request
RETURN-VALUE	O	Value	Description	I32	Return value or status based on the RESPONSE-STATUS. Used to indicate product URI as requestor or responder. Also can be used to provide function call status or error code in case of failed completion.
		1	Product file placed in URI specified by requestor		
		2	Product file placed in URI specified by responder		
		Other	Error code of the failed completion		
Attribute / Mnemonic Information					
ATTRIBUTE-TYPE	R	Value	Description	I16	Indicates the type of Database Attributes for the first mnemonic to be returned.
		6	List of All Mnemonics		
STRUCTURE: Output Product Category Identification					
PROD-NAME	O			String	Name of the product
PROD-DESCRIPTION	O			String	Description of the product in text or xml
PROD-TYPE	O	Value	Description	String	Product type and subtype being requested. (See Table 4-29. Product Categories)
		TAC	Telemetry and Command		
PROD-SUBTYPE	O	DB		String	
NUM-OF-PROD-SUBTYPES	O	1+		U16	Number of further delineations / categories beyond the product subtype. Also, used as msg subject elements ME5, ME6, etc. in Product Message.
PROD-SUBTYPE.n.NAME	O			String	First subcategory of the product subtype. (Subject elements ME5, ME6, etc. of the Product Message)
STRUCTURE: Output File Attributes					
URI	O			String	URI specifying the location where the (single) output file product is stored
NAME-PATTERN	O			String	Describes the name of the output file
DESCRIPTION	O			String	Description of the file in text or xml
FORMAT	O			String	Describes the file format
VERSION	O			String	Identifies the version of the file
SIZE	O	KB		U32	Actual size of the file

For this Attribute-Type the requestor has the option of specifying the URI where the responder should place the product file. If the URI is not specified, the responder will place the product file in its own designated location and return that location in the URI field of the response message. If the requestor specifies the URI, the responder will place the product file in that location, if possible, and

return that same URI from the request message in the response message along with the corresponding RESPONSE-STATUS and RETURN-VALUE values. If the responder is unable to place the product file in the specified URI location, the responder will place the file in an alternate URI location and return the URI in the response message along with the corresponding RESPONSE-STATUS and RETURN-VALUE values. It is possible that the responder cannot access (write to) the URI specified by the requestor, and neither can the requestor access the alternate URI chosen by the responder in which case they will need to work out access and protection issues.

Table 5-116. Meaning of RESPONSE-STATUS and RETURN-VALUE with Recommended Actions

User Specified the URI	RESPONSE-STATUS	RETURN-VALUE	URI	Action
N	Successful	2 (The only meaningful value)	Product was generated and placed in URI chosen by responder	Requestor should retrieve file at responder's URI location
Y	Successful	1	Product was generated and placed in URI specified by requestor	Requestor should retrieve file at the specified URI
Y	Successful	2	Product was generated but placed in alternate URI chosen by responder	Requestor should retrieve file at responder's URI location

The requestor also has the option of specifying the format or version of the product file. If the FORMAT or VERSION is not specified, the responder will use the latest file format or version to build the product. If the requestor specified the FORMAT or VERSION, the responder will generate the product in the desired format or version, if possible. If the responder is unable to generate the product in the format or version requested, the responder will generate the product in the latest format or version along with the corresponding RESPONSE-STATUS and RETURN-VALUE.

5.3.2.3.3 Database Attributes Message Subjects

Table 5-117. Database Attributes Request Message Subject Naming

	Subject Standard	Mission Elements		Message Elements		Miscellaneous Elements	
Subject Element	Specification	MISSION	SAT	TYP	SUBTYP	ME1	ME2
Subject Content	GMSEC	[mission]	[sat]	REQ	DB	[Component: TLM3, TLM2, ...]	
Example for Publisher / Sender	GMSEC	MSSN	SAT1	REQ	DB	TLM3	
Example for Publisher / Sender	GMSEC	MSSN	SAT1	REQ	DB	TLM2	
Example for Subscriber / Receiver	GMSEC	MSSN	SAT1	REQ	DB	>	

Table 5-118. Properties of the *Miscellaneous Elements* for the Database Attributes Request Message

Miscellaneous Element	Required / Optional	Description	Field in Msg, if applicable
ME1	Required	Component name of Responder	NA
ME2	Not used		

Examples:

The TLM2 component (Data Requestor/Subscriber/Client) sends a Database Attributes Request Message to DBMGR (Data Provider/Publisher/Server).

GMSEC.MSSN.SAT1.REQ.DB.DBMGR

DBMGR (Data Provider/Publisher/Server) subscribe subject to receive the Database Attributes Request Message.

**GMSEC.MSSN.SAT1.REQ.DB.DBMGR or
GMSEC.*.*.REQ.DB.DMBGR**

Table 5-119. Database Attributes Response Message Subject Naming

	Subject Standard	Mission Elements		Message Elements		<i>Miscellaneous Elements</i>		
Subject Element	Specification	MISSION	SAT	TYP	SUBTYP	ME1	ME2	ME3
Subject Content	GMSEC	[mission]	[sat]	RESP	DB	[Component: DBMGR, TLM2, ...]	[RESPONSE STATUS: 1-ack,...4-failed]	[]
Example for Publisher / Sender	GMSEC	MSSN	SAT1	REQ	DB	TLM2	1	
Example for Publisher / Sender	GMSEC	MSSN	SAT1	RESP	DB	FD	3	
Example for Subscriber / Receiver	GMSEC	MSSN	SAT1	RESP	DB	*	>	

Table 5-120. Properties of the *Miscellaneous Elements* for the Database Attributes Response Message

<i>Miscellaneous Element</i>	Required / Optional	Description	Field Origination in Msg, if applicable
ME1	Required	Component name of Requestor	Echo of "COMPONENT" in header of Request msg
ME2	Required	Status type supplied by Responder	"RESPONSE-STATUS" from content of Response message

Examples:

Two components, DBMGR (Data Provider/Publisher/Server) and TLM2 (Data Requestor/Subscriber/Client), interact with the Database Attributes Response Message.

DBMGR message subject to send the Databases Attribute Response Message to TLM2 (and another response message to FD):

GMSEC.MSSN.SAT1.RESP.DB.TLM2.1
GMSEC.MSSN.SAT1.RESP.DB.FD.3

TLM2 subscribe subject to receive the Databases Attribute Response Message from DBMGR.

GMSEC.MSSN.SAT1.RESP.DB.TLM2.> or
GMSEC.*.*.RESP.DB.TLM2.>

5.3.3 Satellite Command Messages

The Command Request Message and the Command Response Message are used to send satellite commands and return status between components within the space-ground system.

Command Messages, originating within a missions operation center, transport satellite or spacecraft commands over the ground network for transmission to the satellite. Due to bandwidth narrowing on the uplink that may be considerably less than what is available on the ground, the Command Messages may not be uplinked to the satellite in the same exact format. It is possible that the message as transported through the ground network may be stripped, compacted, or otherwise reduced in volume for transmission to the satellite. Similarly, but in reverse, downlinked data may be expanded or converted from binary or a compacted format into more verbose or descriptive standard message formats. Thus, a bridge task may serve as a middleman / interpreter / converter between messages and data transferred between the ground and satellites.

5.3.3.1 Command Request Message

The Command Request Message is the mechanism to send a satellite command from one component to another within the space-ground system. A satellite command may pass through a number of evolutionary steps that include scheduling, building, creating, validating, transporting, execution, and verification. A satellite command can be input by a flight operations team member through a GUI or command line, or as part of the internal logic of a component. They may be grouped together with processing or execution logic in a file, procedure, or command schedule. The Command Request Message is used to package a command in whatever circumstance it is found and transport it to the next component for processing. Thus, a Command Request Messages may originate from a number of sources such as a GUI / workposition, a command line, a schedule, procedure, or any number of places within a space-ground system. And, it may be used by and pass through a number of components before arriving at its satellite destination.

Table 5-121. Command Request Message Summary

Sender	A GMSEC compliant application responsible for generating or processing a satellite command
Senders Intended Usage	Request processing of a satellite command
Receiver	Application providing a satellite command service such as building, executing, verifying, or transporting
Receivers Intended Usage	Subscribe
What	Action request initiated by user, software, procedure, command schedule, etc.
When	Normally, at scheduled time, or as circumstances warrant
Quality of Service	Guaranteed

Examples:

1. An operator input issuing a satellite command
2. A command schedule execution component
3. A request issued from a procedure

5.3.3.1.1 Command Request Message Information Bus Header

The abbreviated table below shows the required Values of the MESSAGE-TYPE and MESSAGE-SUBTYPE fields for this message.

Table 5-122. Command Request Message Information Bus Header

Field Name	Req/ Opt/API	Value	Type	Notes
Message Information				
HEADER-VERSION	R	2010	F32	Version Number for this message description.
MESSAGE-TYPE	R	REQ	Header string	Message type identifier: REQ, RESP, or MSG
MESSAGE-SUBTYPE	R	CMD	Header string	Unique message identifier, fixed for GMSEC Standard Messages
More ... Please refer to Table 4-9. GMSEC Information Bus Header for a complete definition.

5.3.3.1.2 Command Request Message Contents

Table 5-123. Command Request Message Contents

Field Name	Req/ Opt	Value	Type	Notes
STRUCTURE: Body Content Identification				
CONTENT-VERSION	R	2016	F32	Version Number for this message content description
UNIQUE-ID	A		Header String	
Ground Reference Information				
CMD-SOURCE	O		String	Which user/workposition/proc/schedule the message originated from
Command Information				
CMD-FORMAT	R	[CCSDSPACKET, CCSDSFRAME, CLTU, MNEMONIC, RAW, TDM]	String	Type of command
CMD-DATA	R		Binary	Command data
Ancillary Command Information				
VCID	O		I16	CCSDS Virtual Channel ID
BYPASS	O		Boolean	CCSDS COP-1 flag for "Bypass of Acceptance Check", i.e. without verification
FRAME-COUNTER	O		U32	Reset to next expected command counter
APID	O		String	Application Process Identifier
PACKET-COUNTER	O		U32	
Command Execution Parameters				
CMD-TYPE	O	[REALTIME, FUTURE]	String	If REALTIME, execute upon receipt. If FUTURE, execute at SPACECRAFT-EXECUTION-TIME.
RELEASE-TIME	O		Time	Time the command should begin being released from the front end processor to the remote tracking station.
EARLIEST-UPLINK-TIME	O		Time	Absolute or relative time can apply.
LATEST-UPLINK-TIME	O		Time	Absolute or relative time can apply.
SPACECRAFT-EXECUTION-TIME	D		Time	Required if CMD-TYPE = 'FUTURE'. Absolute or relative time can apply.
Message Response Fields				
RESPONSE	R	Value	Description	Boolean Indicates if a response is required. Defaults to Yes.
		0	False or no response	
		1	True or must respond	

5.3.3.2 Command Response Message

A Command Response Message is sent by an application in response to a Command Request Message. The Command Response Message will provide acknowledgement of the Command Request Message and a status of the action completed. Since the building, transmission, execution, and verification of a satellite command may involve a number of components, the status that is returned may be for any one of these steps in the process.

Table 5-124. Command Request Message Summary

Sender	A GMSEC application that received the Command Request Message
Senders Intended Usage	Reply to the Command Request Message
Receiver	Application that issued the Command Request Message, or an application collecting Command Response Messages for audit trail purposes
Receivers Intended Usage	Subscribe
What	Provide success, failure, or interim status of the progress of the satellite command that was issued/requested
When	Upon receipt of the Command Request Message, or completion of this step of processing
Quality of Service	Guaranteed

Examples:

1. Acknowledge receipt of the satellite command
2. Return status of this step in the sequence of sending a satellite command

5.3.3.2.1 Command Response Message Information Bus Header

The abbreviated table below shows the required Values of the MESSAGE-TYPE and MESSAGE-SUBTYPE fields for this message.

Table 5-125. Command Response Information Bus Header

Field Name	Req/ Opt/API	Value	Type	Notes
Message Information				
HEADER-VERSION	R	2010	F32	Version Number for this message description.
MESSAGE-TYPE	R	RESP	Header string	Message type identifier: REQ, RESP, or MSG
MESSAGE-SUBTYPE	R	CMD	Header string	Unique message identifier, fixed for GMSEC Standard Messages
More ... Please refer to Table 4-9. GMSEC Information Bus Header for a complete definition.

5.3.3.2.2 Command Response Message Contents

Table 5-126. Command Response Message Contents

Field Name	Req/ Opt	Value		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	Echo of the UNIQUE-ID field from the Command request message.
STRUCTURE: Response Status					
RESPONSE-STATUS	R	Value	Description	I16	Identifies the status of the Command being processed
		1	Acknowledgement		
		2	Working/Keep alive		
		3	Successful completion		
		4	Failed completion		
		5	Invalid request		
		6	Final Message		
TIME-COMPLETED	O			Time	Time application completed processing the Command
RETURN-VALUE	O			I32	Return value or status based on the RESPONSE-STATUS. Useful to provide function call status or error code in the case of failed completion
Additional Status and Data					
RELEASE-TIME	O			Time	Time command finished being released from the front end processor to the remote tracking station.
XTCE-STATUS	O	Value	Description	I16	Status codes from the OMG XML Telemetric and Command data Exchange data specification.
		1	Acknowledgement		
		2	Invalid		
		3	TransferredToRange		
		4	SentFromRange		
		5	Received		
		6	Accepted		
		7	Queued		
		8	Executing		
		9	Complete		
10	Failed				
RETURN-DATA	O			Dependent upon response	Additional data that may be desired along with the completion status

5.3.3.3 Command Message Subjects

Table 5-127. Command Request Message Subject Naming

	Subject Standard	Mission Elements		Message Elements		<i>Miscellaneous Elements</i>		
Subject Element	Specification	MISSION	SAT	TYP	SUBTYP	ME1	ME2	ME3
Subject Content	GMSEC	[mission]	[sat]	REQ	CMD	[Component: APP1, TLM2, APP4, TLM3 ...]		
Example for Publisher / Sender	GMSEC	MSSN	SAT1	REQ	CMD	COMMOUT		
Example for Publisher / Sender	GMSEC	MSSN	SAT1	REQ	CMD	ANTENNA5		
Example for Subscriber / Receiver	GMSEC	MSSN	SAT1	REQ	CMD	COMMOUT		

Table 5-128. Properties of the *Miscellaneous Elements* for the Command Request Message

<i>Miscellaneous Element</i>	Required / Optional	Description	Field in Msg, if applicable
ME1	Required	Component name of Responder	NA
ME2 ...	Not used		

Examples:

Two components, CMDEXEC and CMDOUT, interact with the Command Request Message.

CMDEXEC subject to send the Command Request to CMDOUT:

GMSEC.MSSN.SAT1.REQ.CMD.CMDOUT

CMDOUT subject to receive its own Command Request Messages:

GMSEC.MSSN.SAT1.REQ.CMD.CMDOUT

CMDOUT subject to receive any CMDOUT Request Message:

GMSEC..REQ.*.CMDOUT**

CMDOUT subject to receive any kind (REQ or RESP) of Command messages:

GMSEC.*.*.CMD.CMDOUT

Table 5-129. Command Response Message Subject Naming

	Subject Standard	Mission Elements		Message Elements		<i>Miscellaneous Elements</i>		
Subject Element	Specification	MISSION	SAT	TYP	SUBTYP	ME1	ME2	ME3
Subject Content	GMSEC	[mission]	[sat]	RESP	CMD	[Component: TLM3, TLM2...]	[Status]	[Status]
Example for Publisher / Sender	GMSEC	MSSN	SAT1	RESP	CMD	CMDOUT	1	
Example for Publisher / Sender	GMSEC	MSSN	SAT1	RESP	CMD	CMDEXEC	4	
Example for Subscriber / Receiver	GMSEC	MSSN	SAT1	RESP	CMD	CMDEXEC	*	

Table 5-130. Properties of the *Miscellaneous Elements* for the Command Response Message

<i>Miscellaneous Element</i>	Required / Optional	Description	Field Origination in Msg, if applicable
ME1	Required	Component name of Requestor	Echo of "COMPONENT" in header of Request msg
ME2	Required	Status type supplied by Responder	"RESPONSE-STATUS" from content of Response message

Examples:

Two components, CMDEXEC and CMDOUT, interact with the Command Response message.

CMDOUT subject to send the Command Response to CMDEXEC:

GMSEC.MSSN.SAT1.RESP.CMD.CMDEXEC.3

CMDEXEC subscribes to receive its own Command Response Messages:

GMSEC.*.*.RESP.CMD.CMDEXEC.>

5.4 Products and Services Level Messages

In many interactions between components, there is an exchange of data, information, products, or services. This section provides the messages and mechanisms for components to exchange products and services. The messages are generic in nature such that any product or service exchange could use the messages.

5.4.1 Product Messages

5.4.1.1 Product Request Message

A thorough description of this message is given in Section 4.4 Product Messages.

5.4.1.1.1 Product Request Message Information Bus Header

The abbreviated table below shows the required Values of the MESSAGE-TYPE and MESSAGE-SUBTYPE fields for this message.

Table 5-131. Product Request Message Information Bus Header

Field Name	Req/ Opt/API	Value	Type	Notes
Message Information				
HEADER-VERSION	R	2010	F32	Version Number for this message description.
MESSAGE-TYPE	R	REQ	Header string	Message type identifier: REQ, RESP, or MSG
MESSAGE-SUBTYPE	R	PROD	Header string	Unique message identifier, fixed for GMSEC Standard Messages
More ... Please refer to Table 4-9. GMSEC Information Bus Header for a complete definition.

5.4.1.2.1 Product Request Message Content

Table 5-132. Product Request Message Contents

Field Name	Req/ Opt	Value/Description		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	Unique ID used to distinguish the message
Product Input Parameters					
STRUCTURE: Time Window					
START-TIME	O			Time (absolute or relative)	Requested start time for the scope of the product to cover.
STOP-TIME	O			Time (absolute or relative)	Requested stop time for the scope of the product to cover.
Query String					
REQ-STRING	O			String	Specific to the product provider. The string will define a database query, a script expression, Directive string, or some other keyword syntax made known by the provider.
STRUCTURE: Input File Attributes (This Is an array of STRUCTURE: Input File Attributes). Requestor can optionally provide a number of files with the request.					
NUM-OF-INPUT-FILES	O	0+		U16	Indicates the number of files included in this request message.
INPUT-FILE.1.URI	O			String	URI specifying the location where the input file is located
INPUT-FILE.1.NAME-PATTERN	O			String	Describes the file name
INPUT-FILE.1.DESCRPTION	O			String	Description of the file in text or xml
INPUT-FILE.1.FORMAT	O			String	Describes the file format.
INPUT-FILE.1.VERSION	O			String	Identifies the version ID of the file
INPUT-FILE.1.SIZE	O	KB		U32	Size of the included file
STRUCTURE: Data File					
INPUT-FILE.1.DATA	O			Binary (Blob)	The file content
.....
INPUT-FILE.n.URI	O			String	URI specifying the location where the file of the product is stored
INPUT-FILE.n.NAME-PATTERN	O			String	Describes the file name
INPUT-FILE.n.DESCRPTION	O			String	Description of the file in text or xml
INPUT-FILE.n.FORMAT	O			String	Describes the file format.
INPUT-FILE.n.VERSION	O			String	Identifies the version ID of the file
INPUT-FILE.n.SIZE	O	KB		U32	Size of the included file
STRUCTURE: Data File					
INPUT-FILE.n.DATA	O			Binary (Blob)	The file content
STRUCTURE: Output Product Category Identification					
PROD-NAME	O			String	Name of the product
PROD-DESCRIPTION	O			String	Description of the product in text or xml
PROD-TYPE	R	Value	Description	String	Product type being requested. See Table 4-29. Product Categories
		AAA	Archive and Assessment		

		AUTO	Automation		
		FD	Flight Dynamics		
		MAS	Modeling and Simulation		
		PAS	Planning and Scheduling		
		SC	Scripting Control		
		TAC	Telemetry and Command		
PROD-SUBTYPE	R	Product type and subtype being requested. (See Table 4-29. Product Categories)		String	Product subtype being requested. See Table 4-29. Product Categories
NUM-OF-PROD-SUBTYPES	O	1+		U16	Number of further delineations / categories beyond the product subtype. Also, used as msg subject elements <i>ME5</i> , <i>ME6</i> , etc. in Product Message.
PROD-SUBTYPE.n.NAME	O			String	First subcategory of the product subtype. (Subject elements <i>ME5</i> , <i>ME6</i> , etc. of the Product Message)
Product Distribution Information					
RESPOND-VIA-MSG	R	Value	Description	String	Indicates the message to use to deliver the product.
		"MSG.P ROD"	Product Message		
		"RESP.P ROD"	Product Response Message		
STRUCTURE: Product Distribution Options					
DELIVER-VIA-REFERENCE	O	0	No / False	Boolean	Indicates if the product will be referenced by a URI in the message specified above.
		1	Yes / True		
DELIVER-VIA-INCLUDE	O	0	No / False	Boolean	Indicates if the product is to be included in the message specified above.
		1	Yes / True		
STRUCTURE: Output File Attributes					
URI	O			String	Location where the requesting component is asking for the product file(s) to be stored. Could be a web address, directory or folder specification
NAME-PATTERN	O			String	Describes the file name
FORMAT	O			String	Describes the file format
VERSION	O			String	Identifies the version ID of the file
SIZE	O	KB		U32	Maximum size of the file acceptable to the requester. Size specified in KB.

For an explanation on how the START-TIME and STOP-TIME could operate, see Table 5-19. Examples of Start and Stop Times.

5.4.1.2 Product Response Message

A thorough description of this message is given in Section 4.4 Product Messages.

5.4.1.2.1 Product Response Message Information Bus Header

The abbreviated table below shows the required Values of the MESSAGE-TYPE and MESSAGE-SUBTYPE fields for this message.

Table 5-133. Product Response Information Bus Header

Field Name	Req/ Opt/API	Value	Type	Notes
Message Information				
HEADER-VERSION	R	2010	F32	Version Number for this message description.
MESSAGE-TYPE	R	RESP	Header string	Message type identifier: REQ, RESP, or MSG
MESSAGE-SUBTYPE	R	PROD	Header string	Unique message identifier, fixed for GMSEC Standard Messages
More ... Please refer to Table 4-9. GMSEC Information Bus Header for a complete definition.

5.4.1.2.2 Product Response Message Contents

Table 5-134. Product Response Message Contents

Field Name	Req/ Opt	Value/Description		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	
STRUCTURE: Response Status					
RESPONSE-STATUS	R	Value	Description	I16	Identifies the status of the Product Request Message that was processed.
		1	Acknowledgement		
		2	Working / Keep Alive		
		3	Successful Completion		
		4	Failed Completion		
		5	Invalid Request		
		6	Final Message		
TIME-COMPLETED	O			Time	Time application completed processing the request
RETURN-VALUE	O			I32	Return value or status based on the RESPONSE-STATUS. Used to provide function call status or error code in the case of failed completion
STRUCTURE: Output Product Category Identification					
PROD-NAME	O			String	Name of the product
PROD-DESCRIPTION	O			String	Description of the product in text or xml
PROD-TYPE	R	Value	Description	String	Echo of the PROD-TYPE field from the Product Request message.
		AAA	Archive and Analysis		
		AUTO	Automation		
		FD	Flight Dynamics		
		MAS	Modeling and Simulation		
		PAS	Planning and Scheduling		
		SC	Scripting Control		
	TAC	Telemetry and Command			
PROD-SUBTYPE	R	Product type and subtype being requested. (See Table 4-29. Product Categories)		String	Product type and subtype being requested. (See Table 4-29. Product Categories)
NUM-OF-PROD-SUBTYPES	O	0+		U16	Number of further delineations / categories beyond the product subtype. Also, used as msg subject elements <i>ME5</i> , <i>ME6</i> , etc. in Product Message.
PROD-SUBTYPE.n.NAME	O			String	First subcategory of the product subtype. (Subject elements <i>ME5</i> , <i>ME6</i> , etc. of the Product Message)
Output Product Distribution Information					
PROD-MSGs-TO-SEND	O	0+		U16	Indicates the number of Product Msgs that will be published to satisfy the PROD REQ. A value of “-1” can be used to indicate “Unknown”.
Output File Attributes (This Is an array of STRUCTURE: File Attributes)					
NUM-OF-FILES	O	0+		U16	Indicates the number of files included in this response message.
FILE.1.URI	O			String	URI specifying the location where the file of the product is stored
FILE.1.NAME-PATTERN	O			String	Describes the file name
FILE.1.DESCRPTION	O			String	Description of the file in text or xml
FILE.1.FORMAT	O			String	Describes the file format.
FILE.1.VERSION	O			String	Identifies the version ID of the file
FILE.1.SIZE	O	KB		U32	Size of the included file
STRUCTURE: Data File					

FILE.1.DATA	O		Binary (Blob)	The file content
.....
FILE.n.URI	O		String	URI specifying the location where the file of the product is stored
FILE.n.NAME-PATTERN	O		String	Describes the file name
FILE.n.DESCRPTION	O		String	Description of the file in text or xml
FILE.n.FORMAT	O		String	Describes the file format.
FILE.n.VERSION	O		String	Identifies the version ID of the file
FILE.n.SIZE	O	KB	U32	Size of the included file
STRUCTURE: Data File				
FILE.n.DATA	O		Binary (Blob)	The file content

The GMSEC Product Message and Product Response Messages are used for a single product, that is, one product per message. The Product Response and Product Messages allow for multiple files per product.

Table 5-135. Meaning of RESPONSE-STATUS and RETURN-VALUE with Recommended Actions

User Specified the URI	RESPONSE-STATUS	RETURN-VALUE	URI	Action
N	Successful	2 (The only meaningful value)	Product was generated and placed in URI chosen by responder	Requestor should retrieve file at responder's URI location
Y	Successful	1	Product was generated and placed in URI specified by requestor	Requestor should retrieve file at the specified URI
Y	Successful	2	Product was generated but placed in alternate URI chosen by responder	Requestor should retrieve file at responder's URI location
Y or N	Failed	3	Product exceeded maximum requested file size or the default maximum file size	

5.4.1.3 Product Message

5.4.1.3.1 Product Message Information Bus Header

The abbreviated table below shows the required Values of the MESSAGE-TYPE and MESSAGE-SUBTYPE fields for this message.

Table 5-136. Product Message Information Bus Header

Field Name	Req/ Opt/API	Value	Type	Notes
Message Information				
HEADER-VERSION	R	2010	F32	Version Number for this message description.
MESSAGE-TYPE	R	MSG	Header string	Message type identifier: REQ, RESP, or MSG
MESSAGE-SUBTYPE	R	PROD	Header string	Unique message identifier, fixed for GMSEC Standard Messages
More ... Please refer to Table 4-9. GMSEC Information Bus Header for a complete definition.

The Product Message can be used by itself or in conjunction with the Product Request and Product Response Messages. When the Product Message is used by itself, it can contain a notification of product availability or contain the product itself. This is dependent upon the system design and mechanism chosen for product delivery.

The Product Message can also be used with the Product Request and Product Response Messages in one of the two Triad sequences. Some examples of the use of the set of Product Messages are provided in the following table.

Table 5-137. Example Scenarios Using the Set of Product Messages

Service	Message Exchange Pattern	Step 1 Message	Step 2 Message	Step 3 Message
Announce Product Availability	Publish	Product Message:	Option 1: Consumer can retrieve product	
		Contains information about new product	Option 2: Consumer must request product	
Deliver Product Automatically	Publish	Product Message:		
		Contains product		
Deliver Available Product Upon Request	Request Response	Product Request:	Product Response:	
		Consumer requests product	Producer delivers product	
Generate and Deliver Product Upon Request	Triad 1 (Req/ Resp/ Msg)	Product Request:	Product Response:	Product Message:
		Consumer requests product generation and delivery	Producer responds with status, begins product generation	Producer delivers generated product
Announce and Deliver Product Upon Request	Triad 2 (Msg/ Req/ Resp)	Product Message:	Product Request:	Product Response:
		Producer announces product availability	Consumer requests product	Producer delivers product

5.4.1.3.2 Product Message Contents

Table 5-138. Product Message Contents

Field Name	Req/ Opt	Value/Description		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	
Indicator for Last Message in Series					
FINAL-MESSAGE	O	Value	Description	Boolean	When true, indicates the last message in the stream.
		0	No / False		
		1	Yes / True		
STRUCTURE: Response Status					
RESPONSE-STATUS	R	Value	Description	I16	Identifies the status of the Product Message that was processed. Note: Even though a Request is valid, a product may not be able to be successfully generated. In this case the following Product Message would indicate a Failed Completion. Only require for RESP
		1	Acknowledgement		
		2	Working / Keep Alive		
		3	Successful Completion		
		4	Failed Completion		
		5	Invalid Request		
6	Final Message				
TIME-COMPLETED	O			Time	Time application created the product
RETURN-VALUE	O			I32	Return value or status based on the RESPONSE-STATUS. Used to provide function call status or error code in the case of failed completion
Product Distribution Information					
STRUCTURE: Product Distribution Options					
DELIVER-VIA-REFERENCE	O	Value	Description	Boolean	Indicates the product is referenced by a URI. A product can be included in a message, referenced by a URI, or both.
		0	No / False		
		1	Yes / True		
DELIVER-VIA-INCLUDE	O	0	No / False	Boolean	Indicates the product is included in this message. A product can be included in a message, referenced by a URI, or both.
		1	Yes / True		
PROD-MSGS-TO-SEND	O	0+		U16	Indicates the number of PROD Msgs that will be published to satisfy the PROD REQ.
PROD-SEQ-NUM	O	1+		U16	Indicates which message this is in the sequence of PROD Msgs that constitutes a product.
STRUCTURE: Output Product Category Identification					
PROD-NAME	O			String	Name of the product
PROD-DESCRIPTION	O			String	Description of the product in text or xml
PROD-TYPE	R	Value	Description	String	Category of product. Could be echo of the PROD-TYPE field from the Product Request message.
		AAA	Archive and Analysis		
		AUTO	Automation		
		FD	Flight Dynamics		
		MAS	Modeling and Simulation		
		PAS	Planning and Scheduling		
		SC	Scripting Control		
		TAC	Telemetry and Command		
PROD-SUBTYPE	R	Product type and subtype being requested. (See Table 4-29. Product Categories)		String	Subcategory of the product. Could be echo of the PROD-SUBTYPE field of the Product Request Message.

NUM-OF-PROD-SUBTYPES	O	1+	U16	Number of further delineations / categories beyond the product subtype. Also, used as msg subject elements <i>ME5</i> , <i>ME6</i> , etc. in Product Message.
PROD-SUBTYPE.n.NAME	O		String	First subcategory of the product subtype. (Subject elements <i>ME5</i> , <i>ME6</i> , etc.)
Product Properties (This is an array of STRUCTURE: File Attributes)				
NUM-OF-FILES	O	0+	U16	Indicates the number of files included in this response message.
FILE.1.URI	O		String	URI specifying the location where the file of the product is stored
FILE.1.NAME-PATTERN	O		String	Describes the file name
FILE.1.DESCRPTION	O		String	Description of the file in text or xml
FILE.1.FORMAT	O		String	Describes the file format.
FILE.1.VERSION	O		String	Identifies the version ID of the file
FILE.1.SIZE	O	KB	U32	Size of the included file
FILE.1.DATA	O		Binary (Blob)	The file content
.....
FILE.n.URI	O		String	URI specifying the location where the file of the product is stored
FILE.n.NAME-PATTERN	O		String	Describes the file name
FILE.n.DESCRPTION	O		String	Description of the file in text or xml
FILE.n.FORMAT	O		String	Describes the file format.
FILE.n.VERSION	O		String	Identifies the version ID of the file
FILE.n.SIZE	O	KB	U32	Size of the included file
FILE.n.DATA	O		Binary (Blob)	The file content

5.4.1.4 Product Request, Product Response, and Product Message Subjects

Table 5-139. Product Request Message Subject Naming

	Subject Standard	Mission Elements		Message Elements		Miscellaneous Elements	
Subject Element	Specification	MISSION	SAT	TYP	SUBTYP	ME1	ME2
Subject Content	GMSEC	[mission]	[sat]	REQ	PROD	[Component: TLM3, TLM2, ...]	
Example for Publisher / Sender	GMSEC	MSSN	SAT1	REQ	PROD	USER10	
Example for Publisher / Sender	GMSEC	MSSN	SAT1	REQ	PROD	APP5	
Example for Subscriber / Receiver	GMSEC	*	SAT1	REQ	PROD	PLOTGEN	

Table 5-140. Properties of the *Miscellaneous Elements* for the Product Request Message

Miscellaneous Element	Required / Optional	Description	Field in Msg, if applicable
ME1	Required	Component name of Responder	NA
ME2	Not used		

Examples:

Two components, USER10 and PLOTGEN interact with the Product Request Message.

USER10 (Data Requestor/Subscriber/Client) sends a request to PLOTGEN the product Provider/Publisher/Server.

GMSEC.MSSN.SAT1.REQ.PROD.PLOTGEN

PLOTGEN subscribe subject to receive the Product Request Message.

GMSEC.*.*.REQ.PROD.PLOTGEN

GMSEC.MSSN.SAT1.REQ.*.PLOTGEN (PLOTGEN will receive any REQ message)

Table 5-141. Product Response Message Subject Naming

	Subject Standard	Mission Elements		Message Elements		Miscellaneous Elements		
Subject Element	Specification	MISSION	SAT	TYP	SUBTYP	ME1	ME2	ME3
Subject Content	GMSEC	[mission]	[sat]	RESP	PROD	[Component: USER1, TAPS, ...]	[Status]	[]
Example for Publisher / Sender	GMSEC	MSSN	SAT1	RESP	PROD	USER1	1	
Example for Publisher / Sender	GMSEC	MSSN	SAT1	RESP	PROD	SCHED	1	
Example for Subscriber / Receiver	GMSEC	*	SAT1	RESP	PROD	JOE	*	

Table 5-142. Properties of the *Miscellaneous Elements* for the Product Response Message

<i>Miscellaneous Element</i>	Required / Optional	Description	Field Origination in Msg, if applicable
ME1	Required	Component name of Requestor	Echo of "COMPONENT" in header of Request msg
ME2	Required	Status type supplied by Responder	"RESPONSE-STATUS" from content of Response message

Examples:

Two components, the Scheduler (SCHED) (the Data Requestor/Subscriber/Client) and FD (the Data Provider/Publisher/Server) interact with the Product Response Message.

FD sends a response message to the Scheduler (SCHED)

GMSEC.MSSN.SAT1.RESP.PROD.SCHED.1 or
GMSEC.MSSN.SAT1.RESP.PROD.SCHED.4

SCHED subscribes to receive the Product Response Message.

GMSEC.MSSN.*.RESP.PROD.SCHED.> or
GMSEC.MSSN.SAT1.RESP.PROD.SCHED.>

Table 5-143. Product Message Subject Naming

	Subject Standard	Mission Elements		Message Elements		Miscellaneous Elements			
Subject Element	Specification	MISSION	SAT	TYP	SUBTYP	ME1	ME2	ME3	ME4...
Subject Content	GMSEC	[mission]	[sat]	MSG	PROD	[Component of publisher]	[PROD-NAME]	[PROD-TYPE]	[PROD-SUBTYPE]
Example for Publisher / Sender	GMSEC	MSSN	SAT1	MSG	PROD	FD	DAY304	FD	ORBEVT
Example for Publisher / Sender	GMSEC	MSSN	SAT1	MSG	PROD	FD	MAN55	FD	MAN
Example for Subscriber / Receiver	GMSEC	*	SAT1	MSG	PROD	FD	*	FD	ORBEVT

Table 5-144. Properties of the *Miscellaneous Elements* for the Product Message

Miscellaneous Element	Required / Optional	Used For	Description	Field in Msg, if applicable
ME1	Required	Publishing Component	Component name of Publisher	"COMPONENT" from Bus Header of msg
ME2	Required	Product Name	PROD-NAME	The Name of the product
ME3	Required	Product Type or Class	Categorization of the Product type. See Table 4-29. Product Categories.	"PROD-TYPE" from msg content
ME4	Required	Product Subtype or Subclass	Sub-categorization of the Product Type. See above.	"PROD-SUBTYPE" from the msg content
ME5	As necessary	Product Subtype 1	Sub-categorization of the PROD-SUBTYPE	See "ME5 and ME6" note below.
ME6	As necessary	Product Subtype 2	Sub-categorization of the above	See "ME5 and ME6" note below.

ME5 and ME6 Note: The subject elements *ME5*, *ME6*, *ME7* and so on are used to categorize and sub-delineate the products. As many subject elements as necessary can be used to categorize the variety and potentially voluminous number of products. The subscriber may not always know the *ME2* element (PROD-NAME) or the number of subject elements used beyond the basic categorization of product type (*ME3*) and product subtype (*ME4*). In this case, the subscriber should wildcard (*) the *ME2* element and open end (>) the subject elements beyond *ME4*.

Examples:

The Data Provider/Publisher/Server sends out the unsolicited Product Message with the following subject:

GMSEC.MSSN.SAT1.MSG.PROD.FD.ORBEVT.FD.OE

The Requestor/Subscriber/Client subscribes to receive a Product Message categorized with a product type and subtype. It wildcard's the *ME1* and *ME2* fields of component and product name (PROD-NAME), respectively.

GMSEC.MSSN.SAT1.MSG.PROD.*.*FD.OE.> or
GMSEC.MSSN.SAT1.MSG.PROD.FD.ORBEVT.FD.OE.PERAPTIME.>

FD – Component name of product publisher

ORBEVT –PROD-NAME of the product

FD – Product Type (FD = Flight Dynamics)

OE – Product Subtype (OE = Orbital Event)

PERAPTIME - A subtype of **OE**. **PERAPTIME** refers to a product that contains the perigee and apogee times of the orbit.

GMSEC Product Message

Table 5-145. Product Message Subject Name Template

Subject Standard	Mission Elements		Message Elements		Miscellaneous Elements							
Specification	Mission	Satellite	Type	Subtype	ME1	ME2	ME3	ME4	ME5	ME6	ME7	...
"GMSEC"			"MSG"	"PROD"	Producer/ Publisher	Product Name	Product Type	Product Subtype	Product Subtype2	Product Subtype3	Product Subtype4	
GMSEC	tbs	tbs	MSG	PROD	tbs	tbs	*	*				

* Categories of Product Type and Subtypes can be found in Section 4.5.3 Product Categories, or as defined by the mission.

5.4.2 Simple Service Messages

In many service-oriented designs, a one-to-one message exchange will occur between the consumer and producer of a service. That is, the consumer will make a request of the producer of the product or service, and the producer will, in turn, reply with a response. Within the GMSEC architecture, many such message exchanges are possible using a complementary pair of Request and Response type messages. The pair of Simple Service Messages is a general mechanism for the invocation of a service from one component to another. Software components wishing to expose or make certain services available to other components can utilize this general mechanism for that purpose.

The Simple Service Messages are similar in nature and function to the Directive Messages. Where the Directive Request Message will use a keyword or text string to request some functionality of a component, the Simple Service Request Message will make a service request by name, number, and/or operation. It will also pass in any parameters that are required. The Simple Service Messages do not provide for files to be included in the messages though a block of data can be returned in the Simple Service Response message. Also, it is expected that the number of services offered by a component will be small enough that they can be easily identified by name or number.

The intention of the Simple Service messages is to allow a component to offer a small number of simple services to other components and the system in general. They are not meant to provide a comprehensive service framework. Services are expected to be provided locally and not across domains, thereby allowing (or requiring) for all provided services to be uniquely named within the immediate service area. Thus, any component could request any service, by name, offered by another component, if authorized, within the local domain. Since a service name may be used as a subject name element (*ME1*) it must follow the same syntax as elements in a message subject. See Section 3.2 Format of GMSEC Messages Subjects (Topics). Also, see Section 5.2.2.1.3 Simple Service Message Subjects for additional details on service naming.

5.4.2.1 Simple Service Request Message

A Simple Service Request Message is a service request that is issued to one application from another. A service request may also be input from a user through a GUI or command line, or as part of the internal logic of a component. Services could also be grouped together in a procedure (or proc), an executable schedule, or other such orchestration techniques. As components become less coupled, they may tend to offer or provide more services and thus enable more rapid software development with orchestrated modules. The Simple Service Request Message will request the invocation of a single service from a single component.

Table 5-146. Simple Service Request Message Summary

Sender	Any GMSEC compliant application
Senders Intended Usage	Request a function or service
Receiver	Application providing a service, or an application collecting service requests for audit trail purposes
Receivers Intended Usage	Subscribe to requests for services which it will provide
What	Service request initiated by user, software, procedure, or another service, etc.
When	At any time, as necessary
Quality of Service	Guaranteed

Examples:

1. An operator request for information or assistance
2. A request to initiate pre-pass setup
3. Any request issued from a procedure
4. As a provider of an unsolicited service,
 - Automatically provide pass description information at the conclusion of a pass
 - Automatically provide updates to a list
 - Upon some event, provide additional informational data

5.4.2.1.1 Simple Service Request Message Information Bus Header

The abbreviated table below shows the required Values of the MESSAGE-TYPE and MESSAGE-SUBTYPE fields for this message.

Table 5-147. Simple Service Request Message Information Bus Header

Field Name	Req/ Opt/API	Value	Type	Notes
Message Information				
HEADER-VERSION	R	2010	F32	Version Number for this message description.
MESSAGE-TYPE	R	REQ	Header string	Message type identifier: REQ, RESP, or MSG
MESSAGE-SUBTYPE	R	SERV	Header string	Unique message identifier, fixed for GMSEC Standard Messages
More ... Please refer to Table 4-9. GMSEC Information Bus Header for a complete definition.

5.4.2.1.2 Simple Service Request Message Contents

Table 5-148. Simple Service Request Message Contents

Field Name	Req/ Opt	Value	Type	Notes	
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016	F32	Version Number for this message content description	
UNIQUE-ID	R		Header String		
Source Information					
USER	O		String	Which user/workposition/proc/schedule the message is coming from	
STRUCTURE: Service Information					
SERVICE-NAME	O		String	Name of the service offered by a component.	
SERVICE-NUMBER	O		I16	Number of the service.	
SERVICE-VERSION	O		String	Version of the service	
OPERATION-NAME	D		String	Name of the operation within the specified service.	
OPERATION-NUMBER	D		I16	Number of the operation within the specified service.	
OPERATION-VERSION	O		String	Version of the operation within the specified service.	
Service Parameters					
NUM-OF-PARAMS	O		U16	Number of parameters included in the service request	
PARAM.n.NAME	O		String	Name of the parameter	
PARAM.n.VALUE	O		Variable	Value of the parameter Component must ascertain the data type before accessing the value (e.g. with a function call).	
.....	O	
Service Processing Directions					
PRIORITY	O	Value	Description	I16	Indicates processing priority, if applicable
		1	Nominal		
		2	Medium		
		3	High		
RESPONSE	R	Value	Description	Boolean	Indicates if a response is required. Defaults to Yes.
		0	False or no response		
		1	True or must respond		
REQUESTED-EXECUTION-TIME	O		Time		Absolute or relative time can apply.
REQUESTED-EXPIRATION-TIME	O		Time		Absolute or relative time can apply.

Services may evolve over time and services may offer a number of variations or operations for a particular service. A service may be identified by name or number. The requestor may choose either option. If there are a number of operations for that service, then the requestor must specify which operation, by name or number is being requested. When service and operation parameters are not specified, the default will be the first service and first operation within that service as determined by the provider.

The Simple Service Request Message can be used to:

1. Request a service
2. Provide an unsolicited service

To request a service and receive a response via the Simple Service Response Message, the requestor must mark the RESPONSE field as true.

The Simple Service Request Message can also be used to provide an unsolicited service. That is, an application can provide a set of data to other applications automatically, without them having to issue a request. The service provider simply populates the “Parameters” fields of the Simple Service Request Message with the data to be published. The application can also identify the service in the “Service Information” fields. Additionally, the provider of the unsolicited service will publish the message with the name of the service in the *ME1* element of the message subject. Applications wanting to avail themselves of the service will subscribe to the message subject for that service.

As an example, at the conclusion of a pass, an application will automatically provide a description of the pass that includes the start and stop times, the collection point, and the satellite. This data can be inserted into the “Parameter” fields along with the service name, say, ‘PASSDESC”, in the “SERVICE-NAME” field. Also, the “RESPONSE” field is set to false. Then the message is published with ‘PASSDESC” in the *ME1* element of the message subject. Applications wishing to receive this pass description data automatically can easily subscribe to this message and receive all pass descriptions whenever they are published.

Note that for unsolicited services, the Simple Service Request Message will be sent with a “Publish” message exchange pattern.

5.4.2.2 Simple Service Response Message

A Simple Service Response Message is sent by an application in response to a Simple Service Request Message. The Simple Service Response Message will provide acknowledgment of the Simple Service Request Message, a status of the action completed, and any data to be returned. A series of Simple Service Response Messages may be required in the case where the processing of the action is lengthy. An example of this would be a complex mathematical calculation using a large volume of data. In this event, an interim or interactive “working” type message would be issued to let the original application know that the action is still being processed. Please see Section 4.2 GMSEC Messages: Their Characteristics and Interactions for a general discussion on these types of messages.

Table 5-149. Simple Service Response Message Summary

Sender	Application that received the Simple Service Request Message, i.e. the Provider/Producer of the service
Senders Intended Usage	Reply
Receiver	Application that issued the Simple Service Request Message or an application collecting Simple Service Response Messages for audit trail purposes
Receivers Intended Usage	Subscribe for a response to a previously issued service request
What	Provide success/failure response to the service that was requested
When	Upon receipt of Simple Service Request Message or at intervals for those services that are time-consuming
Quality of Service	Guaranteed

Example:

1. Acknowledge receipt of a service request
2. Indicate the Simple Service is still being processed
3. Indicate the Simple Service has completed and include the return status

5.4.2.2.1 Simple Service Response Message Information Bus Header

The abbreviated table below shows the required Values of the MESSAGE-TYPE and MESSAGE-SUBTYPE fields for this message.

Table 5-150. Simple Service Response Information Bus Header

Field Name	Req/ Opt/API	Value	Type	Notes
Message Information				
HEADER-VERSION	R	2010	F32	Version Number for this message description.
MESSAGE-TYPE	R	RESP	Header string	Message type identifier: REQ, RESP, or MSG
MESSAGE-SUBTYPE	R	SERV	Header string	Unique message identifier, fixed for GMSEC Standard Messages
More ... Please refer to Table 4-9. GMSEC Information Bus Header for a complete definition.

5.4.2.2.2 Simple Service Response Message Contents

Table 5-151. Simple Service Response Message Contents

Field Name	Req/ Opt	Value	Type	Notes
STRUCTURE: Body Content Identification				
CONTENT-VERSION	R	2016	F32	Version Number for this message content description
UNIQUE-ID	A		Header String	
STRUCTURE: Response Status				
RESPONSE-STATUS	R	Value	I16	Identifies the status of the Simple Service being processed
		1		
		2		
		3		
		4		
		5		
		6		
TIME-COMPLETED	O		Time	Time application completed processing the Simple Service
RETURN-VALUE	O		I32	Return value or status based on the RESPONSE-STATUS. Provides additional status information particular to the request.
STRUCTURE: Data Abstract				
DATA	O		Dependent upon the service	Additional data that may accompany the response

5.4.2.3 Simple Service Message Subjects

In most request/response message exchange patterns, the *ME1* element is used to identify the requestor and responder of the request. With the Simple Service messages, the *ME1* element may also be used to identify the service. That is, the unique name of the service (following the syntax of the message subject name elements) is inserted into element *ME1*. When the name of the service is inserted into the *ME1* element, the message subject elements TYPE, SUBTYPE, and *ME1* would appear as follows:

... REQ.SERV.[SERVICENAME]
and
... RESP.SERV.[SERVICENAME] ...

The above syntax shows the message subject for a service request message and service response message. In service terminology, the requestor is known as the consumer of the service and the responder is known as the producer or provider. This message subject syntax allows the consumer to request a service without knowledge of the name of the component providing the service. The provider of the service should respond in kind using the service name in the *ME1* element. As should be obvious, the producer must subscribe not only to messages subjects using *ME1* as a component name, but also to message subjects using *ME1* as a service name. *If a single producer provides a large number of services, it will require an equally large number of message subject subscriptions to manage, and this convention may not be desirable.*

Services are expected to be provided locally and not across domains, thereby allowing for all provided services to be uniquely named within the immediate service area. If the service cannot be uniquely named within the service area, then the *ME1* and *ME2* elements can be employed together to uniquely identify all services, similar to the mission-satellite and message type-subtype pairings. The *ME2* element will serve as the general subject matter or group, and the *ME1* element will identify the service, uniquely named, within that group.

Since a service name may be used as a subject name element it must follow the same syntax as elements in a message subject. See Section 3.2 Format of GMSEC Messages Subjects (Topics).

In order to distinguish service names from component names, naming conventions may be established. For example, all services could be named as "S_NNNN", and/or all components could be named "C_NNNN".

Table 5-152. Simple Service Request Message Subject Naming

	Subject Standard	Mission Elements		Message Elements		<i>Miscellaneous Elements</i>		
Subject Element	Specification	MISSION	SAT	TYP	SUBTYP	ME1	ME2	ME3
Subject Content	GMSEC	[mission]	[sat]	REQ	SERV	[Component: APP1, TLM2, TLM3 ...] or [service name]	[service group]	[operation]
Example for Publisher / Sender	GMSEC	MSSN	SAT1	REQ	SERV	APP1		
Example for Publisher / Sender	GMSEC	MSSN	SAT1	REQ	SERV	SERVICEA	GROUPB	OPERATION1
Example for Subscriber / Receiver	GMSEC	*	*	REQ	SERV	SERVICEA	>	

Table 5-153. Properties of the *Miscellaneous Elements* for the Simple Service Request Message

<i>Miscellaneous Element</i>	Required / Optional	Description	Field in Msg, if applicable
ME1	Required	Component name of producer, or name of the service	NA
ME2	Optional	Functional arena, subject matter, or group the service belongs to	NA
ME3	Optional	Name of the operation within the service	OPERATION-NAME
ME4 ...	Not used		

The ME2 and ME3 elements serve as a two element pair to uniquely identify all services, similar to the mission-satellite and message type-subtype pair. Should the name of the service not be unique, the ME2 element can be utilized to avoid ambiguity.

Examples:

Two components, APP4 and APP1, interact with the Simple Service Request Message. APP4 sends a Simple Service Request Message to APP1.

APP4 subject to send a Simple Service Request for a particular service to APP1:

GMSEC.MSSN.SAT1.REQ.SERV.APP1

GMSEC.MSSN.SAT1.REQ.SERV.SERVICEA (using the service name convention in ME1)

GMSEC.MSSN.SAT1.REQ.SERV.SERVICEA.GROUPB (using the optional service name convention of *ME1* and *ME2*)
GMSEC.MSSN.SAT1.REQ.SERV.SERVICEA.GROUPB.OPERATION1 (using all the *ME* elements)

APP1 message subject subscription to receive a request for a particular service using the service name convention in *ME1* (assumes all services are uniquely named):

GMSEC.*.*.*.SERVICEA.>

APP1 message subject subscription to receive a request for a particular service when service names are not unique, using the *ME1* and *ME2* elements.

GMSEC.*.*.*.SERVICEA.GROUPB.>

APP1 message subject subscription to receive a request for a particular operation within a service by using all the subject elements.

GMSEC.*.*.*.SERVICEA.GROUPB.OPERATION1

APP1 message subject subscription to receive requests for any of its provided services when the service naming convention is not used:

GMSEC.*.*.REQ.SERV.APP1

APP1 subjects to receive all requests for any of its services:

GMSEC.MSSN.SAT1.REQ.SERV.APP1.>

GMSEC.MSSN.SAT1.REQ.SERV.SERVICEA.>

GMSEC.*.*.REQ.SERV.SERVICEB.>

GMSEC.MSSN.SAT1.REQ.SERV.SERVICEC.>

and so on for each service.

Table 5-154. Simple Service Response Message Subject Naming

	Subject Standard	Mission Elements		Message Elements		<i>Miscellaneous Elements</i>		
Subject Element	Specification	MISSION	SAT	TYP	SUBTYP	ME1	ME2	ME3
Subject Content	GMSEC	[mission]	[sat]	RESP	SERV	[Component: APP1, TLM2, TLM3 ...] or [service name]	[Status]	[]
Example for Publisher / Sender	GMSEC	MSSN	SAT1	RESP	SERV	APP4	1	
Example for Publisher / Sender	GMSEC	MSSN	SAT1	RESP	SERV	SERVICEA	4	
Example for Subscriber / Receiver	GMSEC	MSSN	SAT1	RESP	SERV	MYAPP	*	

Table 5-155. Properties of the *Miscellaneous Elements* for the Simple Service Response Message

<i>Miscellaneous Element</i>	Required / Optional	Description	Field Origination in Msg, if applicable
ME1	Required	Component name of consumer or name of service	Echo of "COMPONENT" in header of Request msg, or "SERVICE-NAME" in content of Request msg
ME2	Required	Status type supplied by Responder	"RESPONSE-STATUS" from content of Response message

Examples:

Two components, APP4 and APP1, interact with the Simple Service Response message. APP1 sends a Simple Service Response Message to APP4.

APP1 subject to send the Simple Service Response to APP4:

GMSEC.MSSN.SAT1.RESP.SERV.APP4.1 or
GMSEC.MSSN.SAT1.RESP.SERV.SERVICEA.1

APP4 subscribes to receive its own Simple Service Response Messages:

GMSEC.MSSN.SAT1.RESP.SERV.APP4.> or
GMSEC.*.RESP.SERV.APP4.>

5.5 Navigation Data Messages

Within the CCSDS recommended standards there are definitions for various telemetry messages and formats (frames and packets). The GMSEC architecture has defined a corresponding message set to encapsulate the CCSDS telemetry message definitions. (Additionally, GMSEC has defined other types of non-CCSDS telemetry messages.) See Section 5.3.1 Telemetry Data Messages.

Currently, CCSDS has defined a set of six navigation data messages for attitude, orbit, and tracking data. It is expected that these types of definitions will grow in number. The GMSEC architecture will follow a similar course of action with the navigation data messages as it has with the telemetry messages. As a result, the GMSEC architecture will be able to provide support for the existing set of navigation data messages and be extensible in anticipation of accommodating additional navigation data messages.

There are three basic types of CCSDS Navigation Data Messages (NDM).

- Attitude Data Message (ADM)
- Orbit Data Message (ODM)
- Tracking Data Message (TDM)

ADMs are used to convey spacecraft attitude information. These can include:

- Attitude Parameter Message (APM) – Consists of instantaneous attitude state and optional attitude maneuvers
- Attitude Ephemeris Message (AEM) – Consists of a history/forecast of the attitude of the object that can be interpolated to ascertain the attitude of the object at other times

ODMs are used to convey trajectory information. These can include:

- Orbit Parameter Message (OPM) – Consists of a single state vector at a given time that represents the trajectory of the object
- Orbit Mean-Elements Message (OMM) – Consists of a single object at a specified epoch expressed in mean Keplerian elements.
- Orbit Ephemeris Message (OEM) – Consists of a history/forecast of state vectors that can be interpolated to ascertain the trajectory of the object at other times

TDMs are used to convey a variety of tracking data used in the orbit determination process. For example, Doppler and range radiometrics in a variety of tracking modes, very-long-baseline interferometry (VLBI) data, antenna pointing angles, etc.

The CCSDS navigation data messages are summarized in the table below.

Message	Contents	Purpose	CCSDS Document	CCSDS Doc. No.
<i>Attitude Parameter Message (APM)</i>	Attitude state for single object at single epoch	Suitable for exchanges that 1) are automated and/or have human interaction; 2) do not require high fidelity dynamic modeling	Attitude Data Messages	504.0-B-1 (05/2008)
<i>Attitude Ephemeris Message (AEM)</i>	Attitude state for single object at multiple epochs	Suitable for exchanges that 1) automated; 2) require high fidelity dynamic modeling		
<i>Orbit Parameter Message (OPM)</i>	Position and velocity of a single object at a specified epoch	Suitable for exchanges that 1) are automated and/or have human interaction; 2) do not require high fidelity dynamic modeling	Orbit Data Messages	502.0-B-2 (11/2009)
<i>Orbit Mean-Elements Message (OMM)</i>	Specifies orbital characteristics of a single object at a specified epoch	Suitable for exchanges that 1) are automated and/or have human interaction; 2) do not require high fidelity dynamic modeling		
<i>Orbit Ephemeris Message (OEM)</i>	Position and velocity of a single object at multiple epochs within a single time range	Suitable for exchanges that 1) automated; 2) require high fidelity dynamic modeling		
<i>Tracking Data Message</i>	Tracking data for one or more tracking participants at multiple epochs within a specific time range	Convey a variety of tracking data used in the orbit determination process in a single message	Tracking Data Message	503.0-B-1 (11/2007)

CCSDS Navigation messages can be exchanged in one of two formats: keyword value notation (KVN) and XML, the “eXtensible Markup Language”. XML is a better format for specifying the ASCII-based data in the messages. The schema for these types of messages can be found at the CCSDS web site public.ccsds.org in the document “XML Specification for Navigation Data Messages (CCSDS 505.0-B-1, 12/2010). Additionally, other non-CCSDS navigation data messages may be exchanged within the GMSEC architecture. These messages could be in ASCII format, binary, or even a raw tracking data stream.

5.5.1 Attitude Data Messages

Attitude data messages are used to transfer spacecraft attitude information between cooperating entities. They can be used for preflight planning and tracking, tracking and attitude operations, attitude propagations and predictions.

5.5.1.1 Attitude Parameter Message (APM) Contents

The content of the GMSEC Attitude Parameter Message contains a CCSDS APM (or other format) and is used for transferring APMs within the GMSEC architecture. Attitude information within the Attitude Parameter Message contains the state of a single object at a specified epoch. The APM provides information for use in modeling finite maneuvers. Solar radiation pressure can be modeled when accompanied with an Orbit Parameter Message.

The NDM-TYPE field of the Attitude Parameter Message Contents is set to CCSDS-APM or APM for non-CCSDS formats. The NDM-SUBTYPE is a string that describes one of the many types of APMs. The STREAM-MODE field is used to classify the operational mode of the APM. The mode of the APM can be either real-time (RT), replay (RPY), simulation (SIM), or test (TEST). The ACTIVITY-ID is used in conjunction with the mission and vehicle to identify the specific activity that is occurring during the mission. The FORMAT field indicates whether the APM is in XML, keyword value notation, raw, or binary format.

Table 5-156. Attitude Parameter Message Contents

Field Name	Req/ Opt	Value		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	Unique ID used to distinguish the message
Content Body Segment					
Structure: Navigation Data Message Stream Information					
NDM-TYPE	R	[CCSDS-APM, APM]		String	Message contains an Attitude Parameter Message in CCSDS or other format
NDM-SUBTYPE	O	[miscellaneous]		String	Descriptor of the type / kind of the contents of the APM. E.g. Attitude state info, Euler angle rates, or spacecraft parameters
STREAM-MODE	R	Value	Description	String	Identifies the mode of the stream of messages as either Real-time, Replay, Simulator, or Test.
		RT	Real-time		
		RPY	Replay		
		SIM	Simulator		
		TEST	Test/Data Generator		
FINAL-MESSAGE	O	Value	Description	Boolean	When true (and known, especially for replay data), indicates the last message in the stream.
		0	No/False		
		1	Yes/True		
CREATION-TIME	O			Time	Time the Navigation Data Message was created.
Structure: Navigation Data Message Ancillary Information					
ACTIVITY-ID	O			String	Specifies the activity occurring within the mission
ORIGINATOR	O			String	Creating agency. E.g. GSFC-FDF, GSOC, JPL, JAXA etc.
OBJECT-NAME	O				Spacecraft name
OBJECT-ID	O				Spacecraft identifier
Data					
FORMAT	R	[KVN, XML, RAW, BIN]		String	Format of the DATA field KVN: Keyword = Value Notation, XML: eXtensible Markup Language, Raw, or Bin (binary)
DATA	O			String or Binary	Structured Navigation Data Message

5.5.1.2 Attitude Ephemeris Message (AEM) Contents

The content of the GMSEC Attitude Ephemeris Message contains a CCSDS AEM (or other format) and is used for transferring AEMs within the GMSEC architecture. Attitude information within the Attitude Parameter Message contains the state of a single object at a multiple epochs within a specified time range. The AEM provides information for use in dynamic modeling of various kinds of torques such as solar pressure, magnetics, and atmospheric torques.

The NDM-TYPE field of the Attitude Ephemeris Message Contents is set to CCSDS-AEM or AEM for non-CCSDS formats. The NDM-SUBTYPE is a string that describes one of the many types of AEMs. The STREAM-MODE field is used to classify the operational mode of the AEM. The mode of the AEM can be either real-time (RT), replay (RPY), simulation (SIM), or test (TEST). The ACTIVITY-ID is used in conjunction with the mission and vehicle to identify the specific activity that is occurring during the mission. The FORMAT field indicates whether the AEM is in XML, keyword value notation, raw, or binary format.

Table 5-157. Attitude Ephemeris Message Contents

Field Name	Req/ Opt	Value		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	Unique ID used to distinguish the message
Content Body Segment					
Structure: Navigation Data Message Stream Information					
NDM-TYPE	R	[CCSDS-AEM, AEM]		String	Message contains an Attitude Ephemeris Message in CCSDS or other format
NDM-SUBTYPE	O	[miscellaneous]		String	Descriptor of the type / kind of the contents of the AEM. E.g. Quaternion values, spin data, and Euler elements
STREAM-MODE	R	Value	Description	String	Identifies the mode of the stream of messages as either Real-time, Replay, Simulator, or Test.
		RT	Real-time		
		RPY	Replay		
		SIM	Simulator		
		TEST	Test/Data Generator		
FINAL-MESSAGE	O	Value	Description	Boolean	When true (and known, especially for replay data), indicates the last message in the stream.
		0	No/False		
		1	Yes/True		
CREATION-TIME	O			Time	Time the Navigation Data Message was created.
Structure: Navigation Data Message Ancillary Information					
ACTIVITY-ID	O			String	Specifies the activity occurring within the mission
ORIGINATOR	O			String	Creating agency. E.g. GSFC-FDF, GSOC, JPL, JAXA etc.
OBJECT-NAME	O				Spacecraft name
OBJECT-ID	O				Spacecraft identifier
Data					
FORMAT	R	[KVN, XML, RAW, BIN]		String	Format of the DATA field KVN: Keyword = Value Notation,

				XML: eXtensible Markup Language, Raw, or Bin (binary)
DATA	O		String or Binary	Structured Navigation Data Message

5.5.2 Orbit Data Messages

Orbits data messages are used to transfer spacecraft orbit information between cooperating entities. They can be used for preflight planning and tracking, scheduling tracking support, orbit propagation, orbit reconstruction, collision avoidance analysis, and maneuver planning and assessment.

5.5.2.1 Orbit Parameter Message (OPM) Contents

The content of the GMSEC Orbit Parameter Message contains a CCSDS OPM (or other format) and is used for transferring OPMs within the GMSEC architecture. Orbit information within the Orbit Parameter Message contains position and velocity information about a single object for a specific epoch. The OPM provides information for use in modeling maneuvers, atmospheric drag, and other predictive calculations.

The NDM-TYPE field of the Orbit Parameter Message Contents is set to CCSDS-OPM or OPM for non-CCSDS formats. The NDM-SUBTYPE is a string that describes one of the many types of OPMs. The STREAM-MODE field is used to classify the operational mode of the OPM. The mode of the OPM can be either real-time (RT), replay (RPY), simulation (SIM), or test (TEST). The ACTIVITY-ID is used in conjunction with the mission and vehicle to identify the specific activity that is occurring during the mission. The FORMAT field indicates whether the OPM is in XML, keyword value notation, raw, or binary format.

Table 5-158. Orbit Parameter Message Contents

Field Name	Req/ Opt	Value		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	Unique ID used to distinguish the message
Content Body Segment					
Structure: Navigation Data Message Stream Information					
NDM-TYPE	R	[CCSDS-OPM, OPM]		String	Message contains an Orbit Parameter Message in CCSDS or other format
NDM-SUBTYPE	O	[miscellaneous]		String	Descriptor of the type / kind of the contents of the OPM. E.g. state vector, Keplerian elements, maneuvers, matrix
STREAM-MODE	R	Value	Description	String	Identifies the mode of the stream of messages as either Real-time, Replay, Simulator, or Test.
		RT	Real-time		
		RPY	Replay		
		SIM	Simulator		
FINAL-MESSAGE	O	Value	Description	Boolean	When true (and known, especially for replay data), indicates the last message in the stream.
		0	No/False		
		1	Yes/True		
CREATION-TIME	O			Time	Time the Navigation Data Message was created.
Structure: Navigation Data Message Ancillary Information					
ACTIVITY-ID	O			String	Specifies the activity occurring within the mission
ORIGINATOR	O			String	Creating agency. E.g. GSFC-FDF, GSOC, JPL, JAXA etc.
OBJECT-NAME	O				Spacecraft name
OBJECT-ID	O				Spacecraft identifier
Data					
FORMAT	R	[KVN, XML, RAW, BIN]		String	Format of the DATA field KVN: Keyword = Value Notation, XML: eXtensible Markup Language, Raw, or Bin (binary)
DATA	O			String or Binary	Structured Navigation Data Message

5.5.2.2 Orbit Mean-Elements Message (OMM) Contents

Orbital Mean-Elements data messages are used to transfer spacecraft orbital characteristics between cooperating entities. The information can be used to determine future contact parameters between ground and space assets.

The content of the GMSEC Orbital Mean-Elements Message contains a CCSDS OMM (or other format) and is used for transferring OMMs within the GMSEC architecture. Orbital state information within the Orbital Mean-Elements Message can contain mean Keplerian elements, spacecraft parameters, and two line element sets of a single object for a specific epoch. The OMM provides information for use in directing antennas and planning contacts with satellites.

The NDM-TYPE field of the Orbit Parameter Message Contents is set to CCSDS-OMM or OMM for non-CCSDS formats. The NDM-SUBTYPE is a string that describes one of the types of OMMs. The STREAM-MODE field is used to classify the operational mode of the OMM. The mode of the OMM can be either real-time (RT), replay (RPY), simulation (SIM), or test (TEST). The ACTIVITY-ID is used in conjunction with the mission and vehicle to identify the specific activity that is occurring during the mission. The FORMAT field indicates whether the OMM is in XML or keyword value notation.

Table 5-159. Orbital Mean-Elements Message Contents

Field Name	Req/ Opt	Value		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	Unique ID used to distinguish the message
Content Body Segment					
Structure: Navigation Data Message Stream Information					
NDM-TYPE	R	[CCSDS-OMM, OMM]		String	Message contains an Orbit Mean-Elements Message in CCSDS or other format
NDM-SUBTYPE	O	[miscellaneous]		String	Descriptor of the type / kind of the contents of the OMM. E.g. two line element set, Keplerian elements, covariance matrix, or user defined
STREAM-MODE	R			String	Identifies the mode of the stream of messages as either Real-time, Replay, Simulator, or Test.
		Value	Description		
		RT	Real-time		
		RPY	Replay		
		SIM	Simulator		
	TEST	Test/Data Generator			
FINAL-MESSAGE	O			Boolean	When true (and known, especially for replay data), indicates the last message in the stream.
		Value	Description		
		0	No/False		
		1	Yes/True		
CREATION-TIME	O			Time	Time the Navigation Data Message was created.
Structure: Navigation Data Message Ancillary Information					
ACTIVITY-ID	O			String	Specifies the activity occurring within the mission
ORIGINATOR	O			String	Creating agency. E.g. GSFC-FDF, GSOC, JPL, JAXA etc.
OBJECT-NAME	O				Spacecraft name
OBJECT-ID	O				Spacecraft identifier
Data					
FORMAT	R	[KVN, XML]		String	Format of the DATA field KVN: Keyword = Value Notation XML: eXtensible Markup Language
DATA	O			String	Structured Navigation Data Message

5.5.2.3 Orbit Ephemeris Message (OEM) Contents

The content of the GMSEC Orbit Ephemeris Message contains a CCSDS (or other format) OEM and is used for transferring OEMs within the GMSEC architecture. Orbit information within the Orbit Ephemeris Message contains position and velocity information about a single object at multiple epochs within a specific time range. The OEM provides information for use in modeling maneuvers, representing orbits, and other predictive calculations.

The NDM-TYPE field of the Orbit Ephemeris Message Contents is set to CCSDS-OEM or OEM (for non-CCSDS formats). The NDM-SUBTYPE is a string that describes one of the many types of OEMs. The STREAM-MODE field is used to classify the operational mode of the OEM. The mode of the OEM can be either real-time (RT), replay (RPY), simulation (SIM), or test (TEST). The ACTIVITY-ID is used in conjunction with the mission and vehicle to identify the specific activity that is occurring during the mission. The FORMAT field indicates whether the OEM is in XML or keyword value notation.

Table 5-160. Orbit Ephemeris Message Contents

Field Name	Req/ Opt	Value		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	Unique ID used to distinguish the message
Content Body Segment					
Structure: Navigation Data Message Stream Information					
NDM-TYPE	R	[CCSDS-OEM, OEM]		String	Message contains an Orbit Ephemeris Message in CCSDS or other format
NDM-SUBTYPE	O	[miscellaneous]		String	Descriptor of the type / kind of the contents of the OEM. E.g. state vector, Keplerian elements, maneuvers, matrix
STREAM-MODE	R	Value	Description	String	Identifies the mode of the stream of messages as either Real-time, Replay, Simulator, or Test.
		RT	Real-time		
		RPY	Replay		
		SIM	Simulator		
		TEST	Test/Data Generator		
FINAL-MESSAGE	O	Value	Description	Boolean	When true (and known, especially for replay data), indicates the last message in the stream.
		0	No/False		
		1	Yes/True		
CREATION-TIME	O			Time	Time the Navigation Data Message was created.
Structure: Navigation Data Message Ancillary Information					
ACTIVITY-ID	O			String	Specifies the activity occurring within the mission
ORIGINATOR	O			String	Creating agency. E.g. GSFC-FDF, GSOC, JPL, JAXA etc.
OBJECT-NAME	O				Spacecraft name
OBJECT-ID	O				Spacecraft identifier
Data					
FORMAT	R	[KVN, XML]		String	Format of the DATA field KVN: Keyword = Value Notation XML: eXtensible Markup Language

DATA	O		String	Structured Navigation Data Message
------	---	--	--------	------------------------------------

5.5.3 Tracking Data Messages (TDM)

Tracking Data Messages are used to exchange spacecraft tracking data between space agencies, between centers within a space agency, between systems within a center, and other types of interfaces. Some examples of the data within a Tracking Data Message are uplink frequencies, range, Doppler, antenna angles, clock parameters, and meteorological data.

5.5.3.1 Tracking Data Message (TDM) Contents

The content of the GMSEC Tracking Data Message contains a CCSDS (or other type of) TDM and is used for transferring TDMs within the GMSEC architecture. The NDM-TYPE field of the Tracking Data Message Contents is set to CCSDS-TDM or TDM (for non-CCSDS formats). The NDM-SUBTYPE is a string that describes one of the many types of TDMs. Some examples are Doppler and range radiometrics in a variety of tracking modes, very-long-baseline interferometry (VLBI) data, antenna pointing angles, etc. The STREAM-MODE field is used to classify the operational mode of the TDMs. The mode of the TDMs can be either real-time (RT), replay (RPY), simulation (SIM), or test (TEST). The ACTIVITY-ID is used in conjunction with the mission and vehicle to identify the specific activity that is occurring during the mission. The FORMAT field indicates whether the TDM is in XML, keyword value notation, raw, or binary format. The GMSEC Tracking Data Message Contents can optionally contain information about the origin, time, and quality of the data.

Table 5-161. Tracking Data Message Contents

Field Name	Req/ Opt	Value		Type	Notes
STRUCTURE: Body Content Identification					
CONTENT-VERSION	R	2016		F32	Version Number for this message content description
UNIQUE-ID	A			Header String	Unique ID used to distinguish the message
Content Body Segment					
Structure: Navigation Data Message Stream Information					
NDM-TYPE	R	[CCSDS-TDM, TDM]		String	Message contains a Tracking Data Message in CCSDS or other format
NDM-SUBTYPE	O	[miscellaneous]		String	Descriptor of the type / kind of the contents of the TDM. E.g. Doppler, angle, range, one-way.
STREAM-MODE	R	Value	Description	String	Identifies the mode of the stream of messages as either Real-time, Replay, Simulator, or Test.
		RT	Real-time		
		RPY	Replay		
		SIM	Simulator		
		TEST	Test/Data Generator		
FINAL-MESSAGE	O	Value	Description	Boolean	When true (and known, especially for replay data), indicates the last message in the stream.
		0	No/False		
		1	Yes/True		
CREATION-TIME	O			Time	Time the Navigation Data Message was created.
Structure: Tracking Data Message Ancillary Information					
ACTIVITY-ID	O			String	Specifies the activity occurring within the mission
ORIGINATOR	O			String	Creating agency. E.g. GSFC-FDF, GSOC, JPL, JAXA etc.
START-TIME	O			Time (absolute or relative)	Start time of the tracking data
STOP-TIME	O			Time (absolute or relative)	Stop time of the tracking data
FREQUENCY-BAND	O	[C, S, X, Ka, L, UHF]		String	Frequency band for transmitted signals
Data					
DATA-QUALITY	O	[RAW,VALIDATED, DEGRADED]		String	Raw = no quality check Validated = checked and passed Degraded = checked with quality issues.
FORMAT	R	[KVN, XML, RAW, BIN]		String	Format of the DATA field KVN: Keyword = Value Notation, XML: eXtensible Markup Language, Raw, or Bin (binary)
DATA	O			String or Binary	Structured Navigation Data Message

Table 5-162. Navigation Data Message Subject Naming

	Subject Standard	Mission Elements		Message Elements		Miscellaneous Elements				
Subject Element	Specification	MISSION	SAT	TYP	SUBTYP	ME1	ME2	ME3	ME4	ME5
Subject Content	GMSEC	[mission]	[sat]	MSG	NDM	Component of publisher	Stream-mode	NAV-TYPE		
Example for Publisher / Sender	GMSEC	MSSN	SAT1	MSG	NDM	SATSIM	SIM	CCSDS-TDM		
Example for Publisher / Sender	GMSEC	MSSN	SAT1	MSG	NDM	FDF	RT	CCSDS-OPM		
Example for Subscriber / Receiver	GMSEC	*	SAT1	MSG	NDM	*	RT	*		

Table 5-163. Properties of the *Miscellaneous Elements* for the Navigation Data Message

Miscellaneous Element	Required / Optional	Description	Field in Msg, if applicable
ME1	Required	Component name of Publisher	"COMPONENT" from Bus Header of msg
ME2	Required	Identifies stream as real-time, playback, simulator, or test.	'STREAM-MODE' from msg content
ME3	Required	Type of Navigation Data Message	'NAV-TYPE' from the msg content
ME4	Optional	Subtype of Navigation Data Message	NDM-SUBTYPE from msg content or mission specific
ME5	Optional	Activity ID	ACTIVITY-ID from msg content
ME6			

Example for Publisher / Sender of Navigation Data Messages:

GMSEC.MSSN.SAT1.MSG.NDM.CENTER-FACILITY.RT.TDM
 GMSEC.MSSN.SAT1.MSG.NDM.CENTER-FACILITY.RT.CCSDS-TDM
 GMSEC.MSSN.SAT1.MSG.NDM.SATSIM.SIM.CCSDS-OPM
 GMSEC.MSSN.SAT1.MSG.NDM.SATSIM.SIM.CCSDS-OEM

Example for Subscriber / Receiver of Navigation Data Messages:

GMSEC.MSSN.*.MSG.NDM.*.SIM.CCSDS-OPM.>
 GMSEC.MSSN.SAT1.MSG.NDM.*.RT.CCSDS-TDM.>
 GMSEC.MSSN.SAT1.MSG.NDM.ANTENNA5.RT.CCSDS-OEM.EPHEM.ACTY-ID-123

6 GMSEC Services

6.1 Services within the GMSEC Architecture

6.1.1 Description

Initiated in 2001, the Goddard Mission Services Evolution Center (GMSEC) Architecture was designed for missions at the Goddard Space Flight Center (GSFC) to provide a scalable, extensible ground and flight system approach for existing and future mission operation centers. The GMSEC Architecture is a middleware-based, message-oriented communications framework, implemented as the Software Information Bus, also known as the Message Bus. The Message Bus enables the addition, deletion, and exchange of software applications at component-level granularity that are selected to best meet the operational needs of a particular mission.

Using a common Applications Programming Interface (API), standardized messages are published onto the Message Bus (with standardized subjects/topics). Interested components (consumers) subscribe to the standardized subjects to receive and process the published messages. In this manner, producers and consumers of messages are matched up through the publish/subscribe capability of the middleware. The middleware acts as the matchmaker or broker between producers of messages and consumers of messages. Publishers can send messages without regard to the name, whereabouts, and to a great extent, the number of subscribers. Subscribers, with knowledge of the standardized subjects, can receive the published messages with the same general disregard about the publisher.

With the common API, standardized messages & subjects, and the message subject matchmaking of the middleware, a rudimentary service framework has effectively been created. Components can opt to provide services (be Providers) by the (Request) messages they decide to accept and respond to. Components can also avail themselves (be Consumers) of services offered by other components by discovering them at design time and following the message exchange pattern associated with that service. Message exchange patterns (MEPs) are described in the “GMSEC Message Exchange Patterns” section of this document.

Some services may be provided automatically (without being solicited by a consumer) by simply publishing a message with information desired by a consumer. Other services will need to be solicited and a Request/Response MEP will be used. Still other services may require a subscription. In this case, the subscriber(s) must first register with the provider of the service/product. The provider then publishes a series of the service/product/messages over a period of time. Finally, the subscriber can request termination of the service by unregistering with the provider. For each kind of service, the accompanying message exchange pattern must be identified by the provider and followed by the consumer.

While not originally designed specifically as a service-oriented architecture, GMSEC has characteristics and features similar to a SOA. The granularity of provided services is at the component level, not the service level. That is, a software component will typically provide one or more services, grouped as a domain set of functionality. At design time, users must analyze and discover the set of available services provided by a GMSEC-compliant component. GMSEC compliant components, especially legacy components adapted for GMSEC, may have not modified their promotion of capabilities from a functional domain, component-orientation to one of service-orientation. In fact, they may not appreciate that by being GMSEC-compliant they are effectively service providers and consumers.

In this framework, services are contained and managed within each component. This places certain responsibilities upon the service provider. These may include:

- “Advertising” (e.g. publication, electronic database, registry) and describing the service for interested consumers, including the various operations and the interface
- Managing the number of and identity of the consumers
- Managing the volume of service requests as well as the service provision
- Managing the creation, evolution, and deprecation of a service

These service responsibilities should be exercised in accordance with the policies and practices of the overarching governance of the system.

6.1.2 Facets of the GMSEC Software Information Bus

The Software Information Bus or Message Bus consists of

1. Applications Programming Interface (API)
 2. Communications middleware with a (message subject) publish/subscribe mechanism
 3. Standard messages for a variety of mission services/functions
1. The API provides for the exchange of messages, containing data and data products, using a publish/subscribe message exchange format over a variety of common platforms and middlewares. This is accomplished through
 - A set of functions for real-time, asynchronous message transport that normalize the interface and behavior of a number of popular middlewares
 - Message construction/deconstruction functions
 - Subject-based message addressing such that components receive desired messages by subscribing to the appropriate message subjects as published by other components
 2. Publish/subscribe communications
 - Providers of data, products, and services publish information in the form of standardized messages to the Message Bus, compliant with the GMSEC Interface Specification document
 - Consumers of data, products, and services subscribe to messages on the Message Bus
 3. Standardized messages
 - Functions or services can be provided and utilized by exchanging the standardized messages
 - Components can readily communicate without the need for multiple interface control documents. One interface specification to the Message Bus is suitable for all components.

6.1.3 Some Differences between GMSEC Services and other Architectures

1. Services are not identical across architectures

The GMSEC message-based services have been designed independently apart from other service architectures. Therefore, the services that have been identified within each architecture are not identical and do not match on a one-for-one basis. Some services in one architecture will be absent in the other architecture. Also, when services do “match” they will not be identical in their operations or functions. Some services within one architecture will overlap into multiple services of the other architecture.

2. Services need not be requested

Usually in a SOA, services are typically requested. That is, a service is invoked by a consumer, and the result or response is returned by the provider. Within the GMSEC architecture, a service can be solicited or unsolicited. Solicited services will typically be requested with a request/response message exchange. However, some services are provided automatically; with the use of the API publish function. For example, all compliant components will automatically publish Heartbeat messages without being requested to do so. This Heartbeat service of all GMSEC compliant components requires no prodding. A second example is event notification. All components will publish Log messages for public consumption so that interested components can subscribe, analyze, and detect when significant events have occurred. Upon detection of a significant or singular event, a component can kick off contingent processing without having to request notification. Of course other message exchange options are also available for notification or to kick off ancillary processing. Yet a third example is the “advertisement” message. In this circumstance, a component can notify the public with a Log or Product message that a certain product or service is now available. For example, if a schedule product has just finished being compiled & verified, and is now available, a component can offer that product to interested (subscribed) parties with a published Log message.

3. Services vs. messages

Within the SOA framework, a service contains a functional grouping of related operations with each operation uniquely identified. Consumers avail themselves of specific operations within that service. In the GMSEC framework, services that must be solicited are requested through the request/response messages. Various options, or “operations”, for that service are selected by specifying the various optional fields of the Request message. Thus, with one Request message various operations are made available through the selectable options of the message.

4. Publish/subscribe differences

One feature of many high-end middleware products is the publish/subscribe capability. A middleware with this capability will match message subject/topics between publishers and subscribers. GMSEC utilizes the publish/subscribe capability inherent in the middleware to connect service providers and consumers by matching message subjects thereby enabling services within the framework. The middleware acts as a broker to match message subscriptions to messages that are published. Publishers can send messages without regard and subscribers can receive them, even anonymously, by the subject matching performed by the middleware. Publishers and subscribers do not have to know of each other's whereabouts or identity. Within the GMSEC architecture, subscriptions are generally public, with any component able to subscribe to any message. However, additional security options will also allow message publication and subscription to be privatized.

5. Subscriptions

In some service architectures, publishers must first register with a broker application, provide a list of entities it will publish, and then publish all items they are configured to generate to the broker. Similarly, consumers will register with the broker and provide a list of subjects of interest. The broker will match subscribers to publishers and manage the updates to the consumers. The broker can be an intermediary between the publisher and subscriber, or private within the publisher. Here, the matching of subscriptions to published messages is at the application level, either in the intermediary broker, or within the publisher, and not left to the middleware messaging technology as GMSEC does. Also, subscriptions may be private, one-to-one relationships between the broker and consumer.

As stated earlier, a service is provided or obtained by subscription, publication, and exchange of GMSEC-defined messages. Services that can readily be provided by GMSEC-compliant components using the standard GMSEC-defined messages in the GMSEC Interface Specification are listed in the following table.

Table 6-1. GMSEC Services (1 of 2)

Service	Description	Message Exchange Pattern	GMSEC Messages Used
GENERAL SERVICES			
Event Notification	Distribute notification of an event	Publish	Log
Invoke Service	Request a component-provided service	Request/Response	Directive Request Directive Response Simple Service Request Simple Service Response
Provide Service, automatically	Provide a service, unsolicited	Publish	Product Message, Log Message and other "MSG" types
CONFIGURATION AND STATUS REPORTING SERVICES			
Configuration Status Report	Distribute component status	Publish	C2CX Configuration
Device Status Report	Distribute status report of actual or pseudo devices (unsolicited)	Publish	C2CX Device
Heartbeat	Distribute component heartbeat	Publish	C2CX Heartbeat
Resource Status Report	Distribute status report of computer resource usage (unsolicited)	Publish	C2CX Resource
CONFIGURATION CONTROL SERVICES			
Control	Direct a component to action or state change	Publish	C2CX Control
DATA SERVICES			
Real-Time Telemetry Stream	Distribute a real-time telemetry data stream (raw)	Publish	Telemetry Data Messages
Processed Real-Time Telemetry Stream	Distribute a real-time data stream of processed telemetry	Publish	Processed Telemetry Data Messages
Real-Time Parameter Set	Provide a real-time set of parameter values, snapshot or stream	Request/Response (snapshot) Subscription (stream)	Mnemonic Value Request Mnemonic Value Response Mnemonic Value Data
Historic Telemetry Stream	Provide a replay of an historical (archived) telemetry data stream	Subscription	Telemetry Data Messages
Historic Parameter Set	Provide an historic (archived) set of parameter values	Subscription	Mnemonic Value Request Mnemonic Value Response Mnemonic Value Data

Table 6-2. GMSEC Services (2 of 2)

Service	Description	Message Exchange Pattern	GMSEC Messages Used
DATABASE (Telemetry & Command) SERVICES			
List Database Attributes	Provide a list of parameters with their descriptions, limit sets, ...	Request/Response	Database Attributes Request Database Attributes Response
PRODUCT SERVICES			
Announce Product Availability	Announce product is available to access or request	Publish	Product Message or Log Message
Provide Product Automatically (unsolicited)	Distribute product (unsolicited)	Publish	Product Message
Provide Product by Request	Provide a product upon request	Request/Response, Triad 1, or Triad 2	Product Request Product Response [Product Message]
TELECOMMAND SERVICES			
Send Telecommand	Send a command to be uplinked to the satellite	Request/Response	Command Request Command Response
NAVIGATION DATA MESSAGE SERVICES			
Provide NDM Automatically (unsolicited)	Distribute an NDM (unsolicited)	Publish	Any of the NDMs
Provide NDM by Request	Provide an NDM upon request	Request/Response	NDM Request NDM Response

In addition to the above listed services, a mission can easily provide a host of other services using the GMSEC standardized messages. For example, many services could be provided through the standard GMSEC-defined Request and Response Messages using an associated Request/Response message exchange pattern.

Therefore, for application components that provide services, yet retain them within their application, GMSEC has provided a means for providers of those services to make them available, and for consumers to access those services. Thus, GMSEC-compliant components can interact in a SOA-like manner without the rigors of a SOA infrastructure. The following table shows a sampling of services a mission may provide using the standard GMSEC messages.

Table 6-3. Sample Mission Services

Service or Product	Message Exchange Pattern	Consumer Originated Message(s)	Producer Originated Message(s)	Sample Provider/ Producer	Sample Consumer(s)
Activity Schedule Transfer	Request Response	Product Request	Product Response	Mission Planning Manger	Real-time Manager
Ephemeris Data Generation	Publish		Product Message	STK Server	Mission Planning Manager
Generate TLE (Two Line Mean Element)	Request Response	Product Request	Product Response	Ephemeris Provider	Mission Planning Manager
Ground Equipment Status	Publish		C2CX Device	Ground Equipment Client	Ground Equipment Manager
Ground Site Location	Request Response	Product Request	Product Response	Ground Site Provider	Real-time Manager
In-Views Generation	Publish		Product Message	STK Server	Mission Planning Manager
Orbital Events Transfer	Request Response	Product Request	Product Response	Ephemeris Provider	Mission Planning Manager
Out of Limits Notification	Publish		Log Message	Real-time Manager	Post Pass Manager
Pass Description Data	Publish		Log	Real-time Manager	Post Pass Manager
Vehicle Visibility Check	Request Response	Product Request	Product Response	Ephemeris Provider	Real-time Manager

6.1.4 Creating a Service within the GMSEC Architecture

A mission or operation is free to create their own services within a GMSEC-based system. It is recommended that the existing GMSEC messages and message exchange patterns be used whenever possible to facilitate the development and integration effort. However, the provider is not precluded from creating their own customized messages or interface.

When creating a service, the provider and potential users should consider the following:

- What application component will provide the service and where will it execute?
 - How many services overall will this component provide?
- What users (other components) will avail themselves of this service?
 - At what frequency?
 - What loading will be placed on the system from messages and data?
 - What resources will need to be supplied (or managed) to meet the needs of the consumers?
- What existing messages can be used to provide the interface for the service?
 - Directive, Simple Service, Product, or other?
 - As is, or tailored?
 - Is a new or custom message required?
- What message exchange pattern will be required to provide the service? Some options include:
 - The service is automatically provided with no request necessary
 - A consumer must request or initiate the service for each provision and no response is necessary
 - A consumer must request the service in a request/response exchange pattern
 - A consumer must subscribe for the service
- What is the name of the service?
 - What are the various operations available for this particular service?
 - What parameters or data must be provided to the service provider when requesting the service?
 - How should the service and operations be specified in a request? By name and/or number?

6.1.4.1 Using the *ME1* Element of the Message Subject

A service provider can use the *ME1* element of the message subject in a number of ways to provide the service.

1. For many message exchanges, but not all, the *ME1* element specifies the name of the targeted software component. That is, it names the software component the message is destined for. A service provider can require that requestor use the name of the software component that is providing the service in the *ME1* element of the message subject. The service provider will subscribe to all messages destined for it and process them accordingly. This service provision syntax has the benefit of seeing in the message traffic displays and logs where all messages were destined or targeted.
2. A second method of providing the service is to require the name of the service be specified in the *ME1* element. With this syntax, the provider must not only subscribe to messages with its component name in *ME1* but also to messages with the service name in the *ME1* element. With this syntax, the messages will show what services are being requested and provided, but not what components the messages are targeted for.
3. Finally, the *ME1* element could be used with some other syntax to request the service. As above in the second method, it is not obvious which components are exchanging messages by observing or reading the message subjects. In fact, “secret services” could be provided with alphanumeric characters in the *ME1* element that have no obvious meaning. They would need to be interpreted, or the messages examined to determine where they were targeted and what components were subscribing to them.

6.2 Use of GMSEC Services within a Sample Ground System Architecture

This section provides a sample (and simple) ground system architecture to illustrate how GMSEC standard messages might be used to exchange products and services between software subsystems and their components. The sample ground architecture has been organized into a number of smaller subsystems. Within each subsystem the functionality that each subsystem consumes and needs as well as what it produces and provides is identified at a high level. Along with the listed functionality are the GMSEC messages that might be used to acquire or provide that service. The functionality that a subsystem provides is simply defined as data, products, and services.

The list of functions within the subsystems is not definitive or exhaustive. Other architectures and implementations may have more or fewer numbers of subsystems, and distribute the functionality differently. For example, in this sample architecture the data archive is part of the Archive and Assessment subsystem. Some system designs place the telemetry data archive under the Telemetry and Command subsystem, or under the Front-End subsystem. So when a “replay telemetry data” service is requested, the selection, extraction, and replay of previously downlinked and archived telemetry data could be a function of the front-end processor, the telemetry and command subsystem, the data archive, or a combination of these subsystems. Some subsystems will overlap to some degree in their functionality. Other higher level services may require a collaborative effort among subsystems for a system-wide or mission operations service. The list of subsystems in this sample architecture is:

- Flight Dynamics (Navigation and Control)
- Planning and Scheduling
- Archive and Assessment
- Telemetry and Command
- Front End Processors and Simulators
- Automation

Some other subsystems or variations of subsystems that could be part of a subsystem above or even its own subsystem are:

- Maneuver (prediction, control, analysis, product generation)
- Data Processing
- Data Analysis and Modeling
- Data Archive and Data Management
- Product Generation
- Data Distribution
- Communications and Data Transport

- Configuration Monitoring and Control
- Sensor / Instrument / Payload Operations

Other subtleties and differences:

There are a number of ways data, products, and services can be shared and distributed within a system architecture. For example, some products that a subsystem requires may not need to be requested and transferred using the GMSEC Product messages. Rather, a subsystem may be able to access them directly from a known repository.

In this architecture example, the Planning and Scheduling subsystem will do all that is required to produce a daily operating schedule. The Automation subsystem will take on the responsibility of the execution of that schedule.

The Archive and Assessment subsystem in this implementation could be split into separate subsystems such that the Archive subsystem will have the responsibility to archive, manage, and distribute data products. The Assessment subsystem would depend on the Archive subsystem for historical data and the subsequent archiving of any assessment products it produces and is required to manage.

The mission analysis products, their production, maintenance, and management, have been listed in both the Flight Dynamics and Archive and Assessment subsystems.

Maintenance of the telemetry and command database was placed in the Telemetry and Command Subsystem. It could be considered to be part of the Archive and Assessment subsystem.

A number of other variations to the architecture are possible. The subsystems on the following pages are presented in no particular order.

6.1.1 Flight Dynamics Subsystem

Description: Trajectory engineering, attitude determination and control, and mission analysis support

Table 6-4. Flight Dynamics Consumed Services

Consumes/Needs: Data, Products, Services	Services	Message Exchange Pattern	GMSEC Messages
Raw/processed (selective) telemetry data	Real-Time Telemetry Data Stream	Publish	Subscribes To: <ul style="list-style-type: none"> Telemetry Data Message
	Real-Time Parameter Set	Subscription	Publishes: <ul style="list-style-type: none"> Mnemonic Value Data Request Subscribes To: <ul style="list-style-type: none"> Mnemonic Value Data Response Mnemonic Value Data Messages
	Historic Parameter Set	Request/Response	Publishes: <ul style="list-style-type: none"> Archive Mnemonic Value Data Request Subscribes To: <ul style="list-style-type: none"> Archive Mnemonic Value Response [Archive Mnemonic Value Data Messages]
	List Database Attributes	Request/Response	Publishes: <ul style="list-style-type: none"> Database Attributes Request Subscribes To: <ul style="list-style-type: none"> Database Attributes Response
Mission analysis product history	Various mission data products	Request/Response	Publishes: <ul style="list-style-type: none"> Product Request Subscribes To: <ul style="list-style-type: none"> Product Response [Product Message]

Table 6-5. Flight Dynamics Provided Services

Produces/Provides: Data, Products, Services	Services	Message Exchange Pattern	GMSEC Messages
Predicted satellite orbital data and events	Orbital Events, Shadow Times	Request/Response	Subscribes To: <ul style="list-style-type: none"> Product Request Publishes: <ul style="list-style-type: none"> Product Response [Product Messages]
Predicted celestial data and events	Sun-Earth Relationships, Solar Beta Angle	Request/Response	Subscribes To: <ul style="list-style-type: none"> Product Request Publishes: <ul style="list-style-type: none"> Product Response [Product Messages]
Maneuver data, commands, loads	Vehicle Attitude, GPS Navigation Data, Command Sequences, Memory Loads	Request/Response	Subscribes To: <ul style="list-style-type: none"> Product Request Publishes: <ul style="list-style-type: none"> Product Response [Product Messages]
Updates to onboard systems	High Gain Antenna Predictions, Science Field of View Predictions,	Request/Response	Subscribes To: <ul style="list-style-type: none"> Product Request Publishes: <ul style="list-style-type: none"> Product Response [Product Messages]
Acquisition aids (ground & space)	Az/EI Tables, Ground Site Location, Predicted Site Acquisition Table, Vehicle Visibility Check	Request/Response	Subscribes To: <ul style="list-style-type: none"> Product Request Publishes: <ul style="list-style-type: none"> Product Response [Product Messages]
Mission analysis products	Propellant analysis, solar array analysis	Request/Response	Subscribes To: <ul style="list-style-type: none"> Product Request Publishes: <ul style="list-style-type: none"> Product Response [Product Messages]
Special analysis and products	Special Perturbations	Request/Response	Subscribes To: <ul style="list-style-type: none"> Product Request Publishes: <ul style="list-style-type: none"> Product Response [Product Messages]

Note: The above services are a sampling. Some of these services could be provided automatically with just a Product Message, or by request with a Request/Response message exchange pattern, as shown.

6.1.2 Planning and Scheduling Subsystem

Description: Creates optimized near and long term plans for implementation, and generates short term schedules (with associated products) for execution

Table 6-6. Planning and Scheduling Consumed Services

Consumes/Needs: Data, Products, Services	Services	Message Exchange Pattern	GMSEC Messages
Raw/processed (selective) telemetry data for instruments and spacecraft	Real-Time Telemetry Data Stream	Publish	Subscribes To: <ul style="list-style-type: none"> Telemetry Data Message
	Real-Time Parameter Set	Subscription	Publishes: <ul style="list-style-type: none"> Mnemonic Value Data Request Subscribes To: <ul style="list-style-type: none"> Mnemonic Value Response Mnemonic Value Data Messages
	Historic Parameter Set	Request/Response	Publishes: <ul style="list-style-type: none"> Archive Mnemonic Value Data Request Subscribes To: <ul style="list-style-type: none"> Archive Mnemonic Value Response [Archive Mnemonic Value Data Messages]
	List Database Attributes	Request/Response	Publishes: <ul style="list-style-type: none"> Database Attributes Request Subscribes To: <ul style="list-style-type: none"> Database Attributes Response
Satellite orbital data and events	Orbital Events, Shadow Times	Request/Response	Subscribes To: <ul style="list-style-type: none"> Product Request Publishes: <ul style="list-style-type: none"> Product Response [Product Messages]
Celestial data and events	Sun-Earth Relationships, Solar Beta Angle	Request/Response	Subscribes To: <ul style="list-style-type: none"> Product Request Publishes: <ul style="list-style-type: none"> Product Response [Product Messages]
Acquisition aids (ground & space)	Az/EI Tables, Ground Site Location, Predicted Site Acquisition Table, Vehicle Visibility Check	Request/Response	Subscribes To: <ul style="list-style-type: none"> Product Request Publishes: <ul style="list-style-type: none"> Product Response [Product Messages]

Consumes/Needs: Data, Products, Services	Services	Message Exchange Pattern	GMSEC Messages
Routine, daily, special, and long term plans and activities	Science Plans, Activity Plans, Maneuver Plans, Routine Plans	Request/Response	Subscribes To: <ul style="list-style-type: none"> Product Request Publishes: <ul style="list-style-type: none"> Product Response [Product Messages]
Mission analysis product history	Propellant analysis, solar array analysis, other subsystem analysis	Request/Response	Subscribes To: <ul style="list-style-type: none"> Product Request Publishes: <ul style="list-style-type: none"> Product Response [Product Messages]

Table 6-7. Planning and Scheduling Provided Services

Produces/Provides: Data, Products, Services	Services	Message Exchange Pattern	GMSEC Messages
Activity lists and plans	Activity Schedule	Request/Response	Subscribes To: <ul style="list-style-type: none"> Product Request Messages Publishes: <ul style="list-style-type: none"> Product Response [Product Messages]
Command products	Command Sequence Files	Request/Response	Subscribes To: <ul style="list-style-type: none"> Product Request Messages Publishes: <ul style="list-style-type: none"> Product Response [Product Messages]
Loads and images	Payload Schedule, Memory Load, Table Load	Request/Response	Subscribes To: <ul style="list-style-type: none"> Product Request Messages Publishes: <ul style="list-style-type: none"> Product Response [Product Messages]
Pass plans and schedules	Pass Plan, Daily Schedule	Request/Response	Subscribes To: <ul style="list-style-type: none"> Product Request Messages Publishes: <ul style="list-style-type: none"> Product Response [Product Messages]

Note: The above serves are a sampling. Some of these services could be provided automatically with just a Product Message, or by request with a Request/Response message exchange pattern, as shown.

6.1.3 Archive and Assessment Subsystem

Description: 1) Management of the data archive, and 2) health and safety analysis of the spacecraft & mission

Table 6-8. Archive and Assessment Consumed Services

Consumes/Needs: Data, Products, Services	Services	Message Exchange Pattern	GMSEC Messages
Raw/processed telemetry data	Real-Time Telemetry Stream	Publish	Subscribes To: <ul style="list-style-type: none"> Telemetry Data Message
Mission analysis products	Propellant analysis, solar array analysis, other subsystem analysis	Request/Response	Publishes: <ul style="list-style-type: none"> (undefined) "Extract from Archive" Request Message Subscribes To: <ul style="list-style-type: none"> (undefined) "Extract from Archive" Response Message
Telemetry and command database attributes	List Database Attributes	Request/Response	Publishes: <ul style="list-style-type: none"> Database Attributes Request Subscribes To: <ul style="list-style-type: none"> Database Attributes Response

Table 6-9. Archive and Assessment Provided Services

Produces/Provides: Data, Products, Services	Services	Message Exchange Pattern	GMSEC Messages
Raw/processed telemetry	Historic Telemetry Stream, Historic Parameter Set	Request/ Response	Subscribes To: <ul style="list-style-type: none"> • Archive Message Retrieval Request • Replay Telemetry Request • Archive Mnemonic Value Request Publishes: <ul style="list-style-type: none"> • Archive Message Retrieval Response • Replay Telemetry Response • Archive Mnemonic Value Response •
Trending analysis products	Plots, various data formats, reports	Request/ Response	Subscribes To: <ul style="list-style-type: none"> • Product Request Publishes: <ul style="list-style-type: none"> • Product Response • [Product Messages]
General and mission analysis products	Propellant analysis, solar array analysis, payload analysis, other subsystem analysis	Request/ Response	Subscribes To: <ul style="list-style-type: none"> • Product Request Publishes: <ul style="list-style-type: none"> • Product Response • [Product Messages]

6.1.4 Telemetry and Command Subsystem

Description: Uplink and downlink data processing with closed loop control, (including maintenance of the telemetry and command database)

Table 6-10. Telemetry and Command Consumed Services

Consumes/Needs: Data, Products, Services	Services	Message Exchange Pattern	GMSEC Messages
Raw telemetry data (downlink)	Real-Time Telemetry Stream	Publish	Subscribes To: <ul style="list-style-type: none"> Telemetry Data Messages
Telemetry and command database objects and attributes	Real-Time Database Objects	Request/Response	Publishes: <ul style="list-style-type: none"> Product Request Message Subscribes To: <ul style="list-style-type: none"> Product Response Message
Command products, loads and images	Payload Schedule, Memory Load, Table Load	Request/Response	Publishes: <ul style="list-style-type: none"> Product Request Message Subscribes To: <ul style="list-style-type: none"> Product Response Messages
Pass plans and schedules	Pass Plan, Daily Schedule	Request/Response	Publishes: <ul style="list-style-type: none"> Product Request Message Subscribes To: <ul style="list-style-type: none"> Product Response Messages

Table 6-11. Telemetry and Command Provided Services

Produces/Provides: Data, Products, Services	Services	Message Exchange Pattern	GMSEC Messages
Processed telemetry data	Real-Time Parameter Set, Processed Real-Time Telemetry Stream	Subscription, Publish	Subscribes To: <ul style="list-style-type: none"> Mnemonic Value Request Messages Publishes: <ul style="list-style-type: none"> Mnemonic Value Response Message Mnemonic Value Data Messages Processed Telemetry Data Messages
Replay previously archived telemetry data	Historic Telemetry Stream	Request/Response	Subscribes To: <ul style="list-style-type: none"> Replay Telemetry Request Message Publishes: <ul style="list-style-type: none"> Replay Telemetry Response Message Telemetry Data Messages
Command data for uplink	Send Telecommand	Request/Response	Publishes: <ul style="list-style-type: none"> Command Request Messages Subscribes To: <ul style="list-style-type: none"> Command Response Messages
Telemetry and command source database attributes	List Database Attributes	Request/Response	Subscribes To: <ul style="list-style-type: none"> Database Attributes Request Publishes: <ul style="list-style-type: none"> Database Attributes Response

6.1.5 Front End Processors and Simulators Subsystem

Description: Provides interface between ground station and satellite operations ground system

Table 6-12. Front End Processors and Simulators Consumed Services

Consumes/Needs: Data, Products, Services	Services	Message Exchange Pattern	GMSEC Messages
Downlinked telemetry data	Real-Time Telemetry Stream	Publish	Subscribes To: <ul style="list-style-type: none"> Downlinked telemetry packets, frames
Command data for uplink	Send Telecommand	Request/Response	Subscribes To: <ul style="list-style-type: none"> Command Request Messages Publishes: <ul style="list-style-type: none"> Command Response Messages
Telemetry and command database objects and attributes	Real-Time Database Objects	Request/Response	Publishes: <ul style="list-style-type: none"> Product Request Message Subscribes To: <ul style="list-style-type: none"> Product Response Message

Table 6-13. Front End Processors and Simulators Provided Services

Produces/Provides: Data, Products, Services	Services	Message Exchange Pattern	GMSEC Messages
Raw telemetry data, Simulated telemetry data	Real-Time Telemetry Stream	Publish	Publishes: <ul style="list-style-type: none"> Telemetry messages (as received or simulated as downlinked telemetry from the antenna system)
Command data for uplink	Commands for uplink	Publish	Publishes: <ul style="list-style-type: none"> Command data for uplink to the antenna system

Note: The interface between the front end processor and the antenna system may be outside the domain of the operations center, may involve additional interfaces, and may not be service-oriented.

6.1.6 Automation Subsystem

Description: Automatic operation of the system, typically from a single point of control, based upon management policies and procedures

Table 6-14. Automation Consumed Services

Consumes/Needs: Data, Products, Services	Services	Message Exchange Pattern	GMSEC Messages
Notification of and information about events	Event Notification	Publish	Subscribes To: <ul style="list-style-type: none"> Log messages
Periodic “keep alive (heartbeat) messages	Heartbeat	Publish	Subscribes To: <ul style="list-style-type: none"> Component-to-Component Transfer (C2CX) Heartbeat messages
Device, configuration, and resource information	Configuration Status Report, Device Status Report, Resource Status Report	Publish	Subscribes To: <ul style="list-style-type: none"> Component-to-Component Transfer (C2CX) Device, Configuration, and Resource messages
Schedule products for execution	Pass Plan, Daily Schedule	Request/Response	Publishes: <ul style="list-style-type: none"> Product Request Subscribes To: <ul style="list-style-type: none"> Product Response

Table 6-15. Automation Provided Services

Produces/Provides: Data, Products, Services	Services	Message Exchange Pattern	GMSEC Messages
Instructions / directions to orchestrate processes	Invoke Function/Service	Request/Response	Publishes: <ul style="list-style-type: none"> Directive Request Messages Simple Service Request Messages Component-to-Component Transfer (C2CX) Control Messages Subscribes To: <ul style="list-style-type: none"> Directive Response Messages Simple Service Response Messages
Event notification and information	Event Notification	Publish	Publishes: <ul style="list-style-type: none"> Log messages

6.1.7 Common Services across Subsystems

Description: Basic services within the GMSEC Architecture found in all subsystems and components

Table 6-16. Common Consumer Services

Consumes/Needs: Data, Products, Services	Services	Message Exchange Pattern	GMSEC Messages
Instructions to control and direct processing	Invoke Function/Service	Request/Response	Subscribes To: <ul style="list-style-type: none"> • Directive Request Messages • Simple Service Request Messages • Component-to-Component Transfer (C2CX) Control Messages Publishes: <ul style="list-style-type: none"> • Directive Response Messages • Simple Service Response Messages

Table 6-17. Common Provided Services

Produces/Provides: Data, Products, Services	Services	Message Exchange Pattern	GMSEC Messages
(Domain specific products and services are not included here) Notification of and information about events	Event Notification	Publish	Publishes: <ul style="list-style-type: none"> • Log messages
Periodic “heartbeat” (keep alive) messages	Heartbeat	Publish	Publishes: <ul style="list-style-type: none"> • Component-to-Component Transfer (C2CX) Heartbeat messages
Device, configuration, and resource information	Configuration Status Report, Device Status Report, Resource Status Report	Publish	Publishes: <ul style="list-style-type: none"> • Component-to-Component Transfer (C2CX) Device, Configuration, and Resource messages

Appendix A Acronyms

A	API supplied field
ACK	Acknowledge
AOS	Acquisition of Signal
API	Application Programming Interface
APID	Application Process Identifier
C2CX	Component-to-Component Transfer
CCSDS	Consultative Committee for Space Data Systems
CDH	Command and Data Handler
CFG	Configuration
CNTL	Control
COTS	Commercial off the shelf
CVCDU	Coded Virtual Channel Data
D	Dependent
DEV	Device
EU	Engineering Units
EUI	Extended Unique Identifier
GMSEC	Goddard Mission Services Evolution Center
GOTS	Government off the shelf
GSFC	Goddard Space Flight Center
GUI	Graphical User Interface
HB	Heartbeat
ID	Identifier
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet Protocol
IRI	Internationalized Resource Identifier
LOS	Loss of Signal
MAC	Media Access Control
MAL	Message Abstraction Layer
ME	<i>Miscellaneous Element</i>
MEP	Message Exchange Pattern
MOIMS	Mission Operations and Information Management Systems
MSG	Message
NASA	National Aeronautics and Space Administration
O	Optional
OMG	Object Management Group
OS	Operating System
P/S	Publish/Subscribe
R	Required
R/R	Request/Reply or Request/Response
refID	Reference Identification
REQ	Request, Required
RESP	Response
RPY	Replay

RSRC	Resource
RT	Real-Time
SIM	Simulation
SM&C	Spacecraft Monitor and Control
SQL	Structured Query Language
STOL	Spacecraft and Test Operations Language
T&C	Telemetry and Command
TDM	Time Division Multiplexing
UML	Unified Modeling Language
URI	Uniform Resource Identifier
UTC	Universal Time Coordinated
VCID	Virtual Channel Identifier
wrt	With respect to
WSDL	Web Service Description Language
XML	eXtensible Markup Language

Appendix B GMSEC Common Message Structures

Within the GMSEC standard messages, a set of related fields have been used in conjunction with one another and are repeated in a number of messages. These related fields have been grouped together into structures so that they may be easily reused and included within other GMSEC Standard Message. A GMSEC standard message may be composed of a number of structures along with message specific fields as illustrated below.

GMSEC Information Bus Header
HEADER-VERSION
Header Specific Fields
GMSEC Body Content
Structure: Body Content Identification
Structure A
Message Specific Field
Structure B
Message Specific Fields
.
.
.

The following pages depict the structures that have evolved from discussions with the user community. Parameters of similar interest that are also repeated in a number of messages have been formed into a common structure. The structures are then inserted throughout the GMSEC standard messages where appropriate.

B.1 Structure: Body Content Identification

Field Name	Req/ Opt	Value	Type	Notes
STRUCTURE: Body Content Identification				
CONTENT-VERSION	R		F32	Version Number for this message
UNIQUE-ID	A		Header String	Unique ID used by the publisher to distinguish the message

This structure is present in every message and serves to uniquely identify it.

CONTENT-VERSION – is used to identify the version of the message. Most message definitions have evolved and this will serve to let the receiver know what version of the message is to be processed. A receiver may choose to service multiple versions of a message, or only the most recent version of the message.

UNIQUE-ID – Unique ID is generated by the GMSEC API, and used to provide a globally unique identified for the message

B.2 Structure: Body Content Subtype

Field Name	Req/Opt	Value/Description		Type	Notes
STRUCTURE: Body Content Subtype					
C2CX-SUBTYPE	R	Value	Description	Header String	Identifies the type of information being transferred between the Components

Some message definitions contain a number of subtypes to their main type of message. This field will further delineate MESSAGE-SUBTYPE found in the Information Bus Header.

B.3 Structure: Data Abstract

Field Name	Req/ Opt	Value	Type	Notes
STRUCTURE: Data abstract				
DATA	O		Dependent upon usage	

B.4 Structure: Data File

Field Name	Req/ Opt	Value	Type	Notes
STRUCTURE: Data file				
DATA	O		Binary (blob)	The file content

Files are contained within message structures as blobs. This simple structure serves to contain a single file.

B.5 Structure: Directive Keyword

Field Name	Req/ Opt	Value	Type	Notes
STRUCTURE: Directive Information				
DIRECTIVE-KEYWORD	O	Uppercase	Header string	Keyword extracted from the directive string. Useful for routing/processing
DIRECTIVE-STRING	R		String	Full directive string that includes the keyword
SPECIAL-INFO	O		Binary	For application use

A few messages provide the capability for a requester to direct or control another component. One way this can be done is by sending a string of text to be parsed by the receiving component. The string could be specific to the intended component, or a commonly recognized operations language. The syntax of the text string is understood by both parties and worked out at design time.

This structure is also used for the C2CX Control Message that mimics the Directive Request Message.

B.6 Structure: Directive and Service Processing Directions

Field Name	Req/ Opt	Value		Type	Notes
Processing Directions					
PRIORITY	O	Value	Description	I16	Indicates processing priority, if applicable
		1	Nominal		
		2	Medium		
		3	High		
RESPONSE	R	Value	Description	Boolean	Indicates if a response is required. Defaults to Yes.
		0	False or no response		
		1	True or must respond		
REQUESTED-EXECUTION-TIME	O			Time	Absolute or relative time can apply.
REQUESTED-EXPIRATION-TIME	O			Time	Absolute or relative time can apply.

B.7 Structure: File Attributes

Field Name	Req/ Opt	Value/Description	Type	Notes
STRUCTURE: File Attributes				
URI	O		String	Location of the file. Could be a web address, directory or folder specification
NAME-PATTERN	O		String	Describes the name of the output file
DESCRIPTION	O		String	Description of the file contents in text or xml
FORMAT	O		String	Describes the file format
VERSION	O		String	Identifies the version of the file
SIZE	O	Kilobytes	U32	Maximum size of the file acceptable to the requester. Size specified in KB.

When a message contains a file or needs to describe a file, the File Attributes structure is used. Sometimes the structure describes a file to be created and at other times describes an existing file.

Note: If this structure is used more than once in a message definition, a prefix can be appended so all field names will be unique. E.g. fields for an input file could be designated as INPUT-FILE.URI while the output file could be FILE.URI.

B.8 Structure: Mnemonic Selection Criteria

Field Name	Req/ Opt	Value/Description		Type	Notes
STRUCTURE: Mnemonic Selection Criteria					
NUM-OF-MNEMONICS	R	1+		U16	Total Number of mnemonics being requested
MNEMONIC.n.NAME	R	“n” starts at “1”		String	Name of the mnemonic
MNEMONIC.n.DATA-TYPE	O	Value	Description	I16	Indicates the data type to be returned, either the raw value, or the converted value (Engineering Units or Text converted), or both. Defaults to both.
		1	Raw		
		2	Converted		
		3	Both		
MNEMONIC.n.STATE-ATTRIBUTES	O	Value	Description	I16	Indicates if the State Attributes (flags, limits, static flag, and data quality) of the mnemonic are to be returned. Defaults to No.
		1	No		
		2	Yes		
MNEMONIC.n.CRITERIA	O	Value	Description	I16	Identification of when data should be provided for the mnemonic. Includes either upon change of data (value, flags or status), or every sample, or at a specified sampling rate. The default Criteria is “Change” only data.
		1	Change (value, flags, status)		
		2	Every Sample		
		3	Sample Rate		
MNEMONIC.n.SAMPLE-RATE	D	1+	milliseconds	U16	If CRITERIA is specified as “Sample Rate”, this field will specify the data sampling rate for the mnemonic in milliseconds.

Users will need to be able to select and extract data for analysis. This structure provides a means to specify the criteria for selecting data by mnemonic value name.

B.9 Structure: Product Distribution Options

Field Name	Req/Opt	Value/Description		Type	Notes
STRUCTURE: Product Distribution Options					
DELIVER-VIA-REFERENCE	O	Value	Description	Boolean	Indicates if the data will be referenced by a URI in the single response message. Defaults to No.
		0	No / False		
		1	Yes / True		
DELIVER-VIA-INCLUDE	O	0	No / False	Boolean	Indicates if the data is to be included in the single response message. Defaults to Yes.
		1	Yes / True		

Requestors of a product have the option of specifying how they should be delivered. Delivery can be:

- By reference – provide a location where the product can be accessed
- Include – include the product file within the Response message
- Both

B.10 Structure: Product Category Identification

Field Name	Req/Opt	Value/Description		Type	Notes
STRUCUTRE: Product Category Identification					
PROD-NAME	O			String	Name of the product being requested
PROD-DESCRIPTION	O			String	Description of the product in text or xml
PROD-TYPE	O	Value	Description	String	Product type and subtype being requested. (See Table 4-29. Product Categories)
PROD-SUBTYPE	O	MSG	Message	String	
NUM-OF-PROD-SUBTYPES	O			U16	Number of further delineations / categories beyond the product subtype. Also, used as msg subject elements <i>ME5</i> , <i>ME6</i> , etc. in the Product Message.
PROD-SUBTYPE.n.NAME	O			String	First subcategory of the product subtype. (Subject elements <i>ME5</i> , <i>ME6</i> , etc. of the Product Message)

In order to describe the type and kind of a product, a general hierarchical classification of the products produced by the various space-ground systems has been created. When a product is published or included in a message, the name and type of the product can be described. Hence, users can subscribe to classes or types or names of products, as well as be assured of the kind of product in a message, as it has been described.

B.11 Structure: Response Status

Field Name	Req/Opt	Value/Description		Type	Notes
STRUCTURE: Response Status					
RESPONSE-STATUS	R	Value	Description	I16	Identifies the status of the Request Message that was processed.
		1	Acknowledgement		
		2	Working / Keep Alive		
		3	Successful Completion		
		4	Failed Completion		
		5	Invalid Request		
		6	Final Message		
TIME-COMPLETED	O			Time	Time application completed processing the request
RETURN-VALUE	O			I32	Return value or status based on the RESPONSE-STATUS. Used to provide function call status or error code in the case of failed completion

The Response Status structure found in Response messages returns the status of the Request. The RESPONSE-STATUS field returns both successful and unsuccessful status. Therefore, the Response message can serve as a normal message as well as a fault message. The RESPONSE-STATUS values are fully explained in Section 4 of this document. The RETURN-VALUE will supply additional information for analysis should the request fail.

B.12 Structure: Satellite Command Description

Field Name	Req/ Opt	Value	Type	Notes
STRUCTURE: Satellite Command Description				
CMD-FORMAT	R	[CCSDSPACKET, CCSDSFRAME, CLTU, MNEMONIC, RAW, TDM]	String	Type of command
CMD-DATA	R		Binary	Command data

B.13 Structure: Satellite Command Execution Time Parameters

Field Name	Req/ Opt	Value	Type	Notes
STRUCTURE: Satellite Command Execution Parameters				
CMD-TYPE	O	[REALTIME, FUTURE]	String	If REALTIME, execute upon receipt. If FUTURE, execute at SPACECRAFT-EXECUTION-TIME.
RELEASE-TIME	O		Time	Time the command should begin being released from the front end processor to the remote tracking station.
EARLIEST-UPLINK-TIME	O		Time	Absolute or relative time can apply.
LATEST-UPLINK-TIME	O		Time	Absolute or relative time can apply.
SPACECRAFT-EXECUTION-TIME	D		Time	Required if CMD-TYPE = 'FUTURE'. Absolute or relative time can apply.

B.14 Structure: Satellite Command Parameters

Field Name	Req/ Opt	Value	Type	Notes
STRUCTURE: Satellite Command Parameters				
VCID	O		I16	CCSDS Virtual Channel ID
BYPASS	O		Boolean	CCSDS COP-1 flag for "Bypass of Acceptance Check", i.e. without verification
FRAME-COUNTER	O		U32	Reset to next expected command counter
APID	O		String	Application Process Identifier
PACKET-COUNTER	O		U32	

B.15 Structure: Telemetry Data Stream Information

Field Name	Req/ Opt	Value/Description		Type	Notes
STRUCTURE: Telemetry Data Stream Descriptors					
FORMAT	R			String	Message contains CCSDS packet or frame
COLLECTION-POINT	O			String	Receiver, device, point, path, etc. where data was received. Used to distinguish data simultaneously received at multiple collection points.
STREAM-MODE	R	Value	Description	String	Identifies the kind or source of the stream of telemetry as either Real-time, Replay, Simulator, or Test.
		RT	Real-time		
		RPY	Replay		
		SIM	Simulator		
		TEST	Test/Data Generator		
FINAL-MESSAGE	O	Value	Description	Boolean	When true (and known, especially for replay data), indicates the last message in the stream.
		0	No/False		
		1	Yes/True		
LENGTH	O	Bytes		U32	Length of frame or packet
TIME	O			Time	Time of frame, usually ground receipt time

This structure describes telemetry data streams.

The STREAM-MODE serves to describe the type and implied source of the telemetry data stream.

- RT – real-time from the satellite
- RPT – a replay of real archived data previously downlinked
- SIM – data from a satellite simulator
- TEST – simple test data without the capabilities of a simulator

B.16 Structure: Telemetry Data Channel Information

Field Name	Req/ Opt	Value/Description	Type	Notes
STRUCTURE: Telemetry Data Channel Information				
PHY-CHAN	R		String	Physical channel on which data is received
VCID	R		I16	Virtual Channel ID

This structure describes telemetry data channels.

B.17 Structure: Time Window

Field Name	Req/Opt	Value/Description	Type	Notes
STRUCTURE: Time Window				
START-TIME	R		Time (absolute or relative)	Requested start time of the messages to be retrieved from the Message Archive
STOP-TIME	O		Time (absolute or relative)	Requested stop time of the messages to be retrieved from the Message Archive. Defaults to the end of the Message Archive.

A number of request messages require a time window when specifying data selection criteria. Depending on the request and data desired, either the start time, stop time, or both are required. Rarely, if ever, will both be optional.

Appendix C GMSEC Application Programming Interface

C.1 Messaging Overview

Within the GMSEC architecture, software components serve as functional units for the mission operations control system. Each component is supported by a platform, encompassing the processing hardware, operating system and supporting software libraries. Components can share the same platform, or not, as allowed by component platform requirements and as elected by the system designer.

Middleware is defined as an underlying communications package, or software backplane service that provides communication services between applications. It isolates much of a component's complexity from other components as well as unburdens the application from having to program the details of interprocess communication. Communication between applications is in the form of messages that are transferred from application to application by the middleware. The GMSEC Application Programming Interface (API) is an intermediate layer between the component and the middleware. The GMSEC API serves to provide a standard functional interface for the component to a number of available and compatible underlying middlewares. Application components maintain a connection to the GMSEC API which in turn communicates with the underlying middleware to provide a broad set of communication services.

The application's connection to the GMSEC API and middleware can be implemented as an integrated part of the application design, as an interface adaptor, or as an external software component designed to mediate between the application component's native interface and the GMSEC standard. The following terms help clarify the various methods of interfacing with the GMSEC API.

- **GMSEC API** – the specification of the interface and its behavior. Also, refers to the software implementation.
- **GMSEC Bus** (AKA “The Bus, “The Message Bus) - The combination of the GMSEC API, underlying middleware, and the standardized messages
- **Adaptor** - A convertor from the native application to the GMSEC message bus and the GMSEC message format (compiled with the application)
- **Proxy** – An external application serving the role of adaptor between an application and the GMSEC bus (compiled externally from the application)
- **Wrapper** - A set of code either enabling an application to run in a non-native environment, or to enable an application to interact with a non-native interface

Figure C.1-1 illustrates connections between applications and middleware components in the GMSEC Architecture.

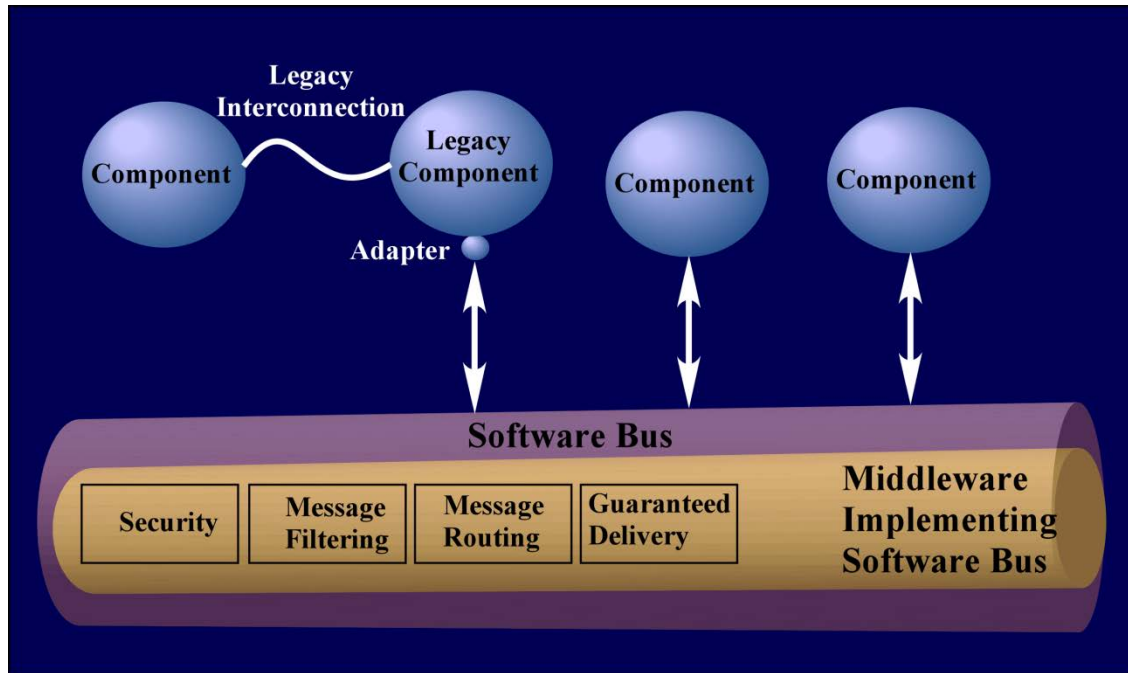


Figure C.1-1. Middleware Overview

Applications communicate with one another through their GMSEC interface (via the underlying middleware) using messages. Each of these messages includes a specific subject that defines both the content and meaning of the message. Applications send messages by providing the message and message subject to the GMSEC API which in turn provides it to the middleware. The middleware takes the responsibility of routing a copy of any message with that subject that appears on the software bus to the requesting application(s). Applications receive messages by providing (subscribing to) the requested message subject via the GMSEC API to the middleware. This subject-based message addressing means message producers need not know the location, quantity, or platform of message consumer(s).

C.1.1 Publish / Subscribe

In Publish/Subscribe interactions, data producers are decoupled from data consumers. That is, the publishers are typically unaware of the location and number of data consumers. The data is published for any consumer that subscribes to the specified subject. No coordination or protocol is required between the publisher and subscriber except for the named subject. In fact, any number of publishers from any number of sources could provide the data on a unique subject to any number of subscribers at any number of destinations. The following table provides examples of how this mechanism might be used.

Table C-1. Publish/Subscribe Scenarios

Publishers	Subscribers	Example Scenario
One	One	A response to an application's specific request for data
Many	One	An archive application may be the central point for collecting data from multiple sources.
One	Many	Data value updates might be distributed to many applications monitoring the same data point(s).
Many	Many	Many applications publish log messages that are also being monitored (subscribed to) by many displays on various workstations.

Underlying middleware products could publish a message for each subscribed recipient, or make use of a multicast mechanism where one message is published over the network that can be received by a multiple number of subscribers. (The multicast term is a misnomer. There is only a single "cast" of a message, but multiple receivers.)

C.1.2 Request / Reply

The middleware will also support session-based communications between components, allowing an application to reply to a specific message. Two applications can find and interact with each other privately across the middleware framework. This mechanism allows the emulation of existing socket-based inter-applications communications in the GMSEC system.

Request/Reply interactions will occur between two cooperating applications, a request sender and a message receiver. The receiver will issue a reply to the requester. The receiver of the message will have inherent knowledge about the received message in order to process it and provide the requested information, if any, in the reply. (Note that a reply could simply be an acknowledgement.) That is, it is assumed that a prearranged protocol has been established for the Request/Reply message interaction. The point-to-point connection will use a unique subject name previously agreed upon and associated with the request and reply messages that will effectively guarantee the point-to-point message exchange.

The point-to-point connection between the requester and receiver need not be established prior to an attempt to send a request message. If an application attempts to send a request message and the receiving application has not subscribed to the subject and there is no persistence on the connection, then the message will not be delivered and an error condition will be returned to the sending application.

C.1 3 Persistence

Connections or communication paths between cooperating applications sometimes require a delivery mechanism that goes beyond what is provided with a reliable delivery quality of service. Persistence is the name for a delivery mechanism that allows delivery of messages beyond temporary network outages and program restarts. The GMSEC API provides whatever data persistence is supplied by the underlying middleware.

C.2 Application Programming Interface (API)

The GMSEC application programming interface (API) specification has been developed for a variety of programming languages to allow application and adaptor developers choices. The API has been designed around the idea of calls and callbacks (in the procedural sense) or object method invocations (in the object-oriented sense).

Each application would use a connect service to initiate communications with the middleware via the GMESC API, and then use the Publish/Subscribe/Request/Reply services to perform their communications with other applications. The middleware would handle the routine sending of messages; delivery and queuing of messages, and subject based filtering for the application.

A diagram showing the software layers between the application components, standardized GMSEC messages, API, and operating systems is show in Figure C.2-1 GMSEC Interface Layers.

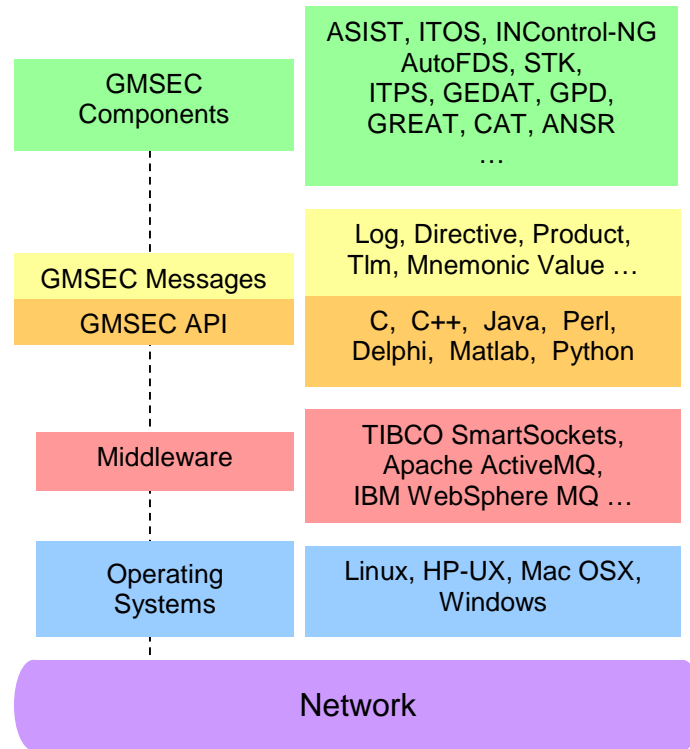


Figure C.2-1. GMSEC Interface Layers

C.2.1 API Function Descriptions

Descriptions of the API functions are organized by class. The major classes of the API are:

- **ConnectionFactory** – a static level interface to manage the GMSEC bus connection
- **ConnectionConfig** – used by the ConnectionFactory to provide a generic means of supplying initialization data without being middleware specific
- **Connection** – provides message transport services on the GMSEC connection
- **Message** – a container that provides services to construct and manipulate messages
- **Field** – a container that provides services to construct and manipulate fields within a message. Has a subclass for each primitive type.
- **Status** – provides status feedback
- **Callback** – provides a means for the applications to process inbound messages in an asynchronous manner

The following text provides a very high level description of some of the functions within the major classes.

Connection Functions

- **Connect**: Initiate the connection between the application and the middleware
- **Disconnect**: Terminate the connection between the application and the middleware

Publish/Subscribe Functions

- **Publish** <message>: Send a message to all subscribers on the previously established connection
- **Subscribe** <subject, [callback]>: Registers a subject name to be received by the connection and will optionally associate a callback to be called when messages are received on the given subject
- **Unsubscribe** <SubscriptionInfo>: Removes the subscription which corresponds to the SubscriptionInfo.

Request/Reply Functions

- **Request** <requestmessage, timeout, replymessage, [replycallback]>: Sends a request message on the connection and will handle the reply (a response message). This function will block until the reply is received or the timeout expires. When a callback is supplied, it will be called when the reply (response message) is received.
- **Reply** <message, replymessage>: Used by the receiver of a request message to send a response message back to the sender.

Message Retrieval Functions

- **Receive** <message, timeout>: Used by a subscriber to pull the next received message from the incoming queue. Timeout determines how long to wait in case no messages are received.
- **DispatchMsg** <message>: Used by a subscriber to cause the callback for a received message to be invoked.
- **StartAutoDispatch** <connection>: Begin asynchronous delivery of messages through a separate threading mechanism. This is an alternative to calling GetNextMsg. Messages received by the connection will be delivered to the registered callbacks by the dispatcher.
- **StopAutoDispatch** <>: Terminates asynchronous delivery of messages by a separate threading mechanism.

Each application would use the GMSEC API connect service to establish communications with the middleware, and then use the publish/subscribe and request/reply services for passing messages. Other functions not shown facilitate the creating of a message's contents to allow an application to build up the content of a message, field-by-field, prior to shipping. Depending upon the

needs of the application, incoming messages can be received, dispatched and processed in a number of ways with the message retrieval functions.

C.3 Ancillary Programming Information

C.3.1 Programming Languages and Platforms Supported

The GMSEC API has been developed for a limited set of computer platforms, in a number of software development languages, and for use with a limited number of underlying middleware vendors. The supported languages and platforms are determined and driven by the requirements and needs of the GMSEC user community, and will therefore represent the most popular or commonly used. The table below shows the languages, platforms, and middleware for which the API has been implemented. Additional platforms and interfaces could be supported in the future.

Table C-2. API Languages, Platforms, and Middleware

	Supported Interfaces
Supported Languages	<ul style="list-style-type: none"> • C, C++ • Java • Perl <ul style="list-style-type: none"> ◦ Linux, Solaris: CPAN ◦ Windows: ActiveState (ActivePerl)
Operating Systems	<ul style="list-style-type: none"> • Linux • Macintosh • Microsoft Windows • HP-UX
Compilers	<ul style="list-style-type: none"> • Linux: GNU C++ (GCC), Java, Perl • OSX: Xcode GCC, Java, Perl • Windows: Visual Studio for C++, Java, ActivePerl
Middleware	<ul style="list-style-type: none"> • Apache ActiveMQ • GMSEC Message Bus, GMSEC Bolt • IBM WebSphere MQ • Oracle WebLogic, Oracle Fusion M/W • TIBCO SmartSockets

For specific versions of operating systems, compilers, and middleware, please refer to the API User's Guide.

C.3.2 Libraries

The GNU Common C++ library is being used to handle cross-platform issues such as threading. It can be found at <http://www.gnu.org/software/commonc++/>.

C.3.3 Building and Linking Applications

This information can be found bundled with the API software referenced in Section C.2.1 API Function Descriptions.

Appendix D Future Considerations

This section records potential changes to this document. The changes may include document structure and/or content.

1. Content
 - a. Security
 - i. Add Login and Logout (authentication) messages to Control and Monitor category
 - ii. Add messages and mechanism for lock and key, or digital certificates
 - b. In addition to the three existing GMSEC message types of REQ, MSG, and RESP add the following message types. They would complement the CCSDS SM&C MAL patterns of interaction.
 - i. ERR message type.

This would complement the required error message when an operation fails. It would use the GMSEC defined Response messages with the appropriate error code in the RESPONSE-STATUS field of the message.
 - ii. ACK message type.

This would complement the ACK message required for a number of MEP. It would use the GMSEC defined Response messages with a status code of "Acknowledgement" in the RESPONSE-STATUS field of the message.
 - c. Should Command Request Message allow input of a file?
 - d. Add messages to request storage and retrieval of a product or file into an archive or database ("Store This", "Retrieve This")
 - e. Change data structures to include arrays

Appendix E Checklist for Updating This Document

This section contains a list of things to check and/or update when making a change to this document.

Document Changes

Check ✓	Section or Area	Reference	Description
	0	Cover Page	- Version # - Release Date
	0	Table 0-1. Version History of the Interface Specification	Add entry to table for each version
	0	Table 0-2. Change Information	- Update with changes for this version.
	1	1.3.2 GMSEC Compliant Components	Ensure following reference is valid: "Please see Section 3.7 GMSEC Interoperability in the GMSEC Architecture Document (LATEST RELEASE VERSION) for a discussion on what it means to be GMSEC compliant."
	1	1.4.1 GMSEC Documents	Update release version and date for referenced documents.
	5	5.1.2 Message Definition History	- Table 5-1. GMSEC Message Definitions History, 1 of - Table 5-2. GMSEC Message Definitions History, 2 of - Table 5-3. GMSEC Message Definitions History, 3 of - Table 5-4. GMSEC Message Definitions History, 4 of 4 - Ensure for each message (row): <ul style="list-style-type: none">• Version # of message• Cross ref. to document section• Cross ref. to page #
	C	C.2 Application Programming Interface (API) - Figure C.2-1 GMSEC Interface Layers	Review middleware and component names to ensure they are not obsolete
	C	C.3.1 Programming Languages and Platforms Supported - Table C-2 API Languages, Platforms and Middleware	Ensure content of table is current

For a New or Modified Message

Check √	Section or Area	Reference	Description
	3	Table 3-1. Sample Subject Filtering Items in Addition to Type and Subtype	Update for new messages, as desired
	3	Table 3-8. Descriptions of the Message Elements of the GMSEC Message Subject	Update for new messages
	3	3.4 Definitions of GMSEC Subject Names	- Update tables for new messages - Update tables if a message subject changes
	3	Table 3-15. GMSEC Message Subject Definitions, 4 of 4	4 tables
	4	Table 4-27. GMSEC Message Functional Grouping	Update for new messages
	4	Table 4-28. Software Class and Subclass Categories	- Update for new classes or redefinition of existing classes - Update for new or obsolete components
	5	Table 5-1. GMSEC Message Definitions History, 1 of 4	Ensure the "HEADER-VERSION" field of each message agrees with the value in the column "Last Defined or Modified Version" – ALL TABLES
	5	Table 5-2. GMSEC Message Definitions History, 2 of 4	Ensure the "CONTENT-VERSION" field of each messages agrees with the value in the column "Last Defined or Modified Version" – ALL TABLES
	5	Table 5-3. GMSEC Message Definitions History, 3 of 4	
	5	Table 5-4. GMSEC Message Definitions History, 4 of 4	
	5	Table 5-15. Examples of Time Fields in GMSEC Messages	Update, as appropriate, for a new message or modified message
	5	Table 5-16. Examples of Time Fields in GMSEC Messages,	Update, as appropriate, for a new message or modified message

